7-28-1975

Computer Aided Design for Fluidic Sequential Circuits of Fundamental Mode

Yau-Hwang Lee
Portland State University

AN ABSTRACT OF THE THESIS OF Yau-Hwang Lee for the Master of Science in Applied Science presented July 28, 1975.

Title: Computer Aided Design for Fluidic Sequential Circuits of Fundamental Mode.

APPROVED BY MEMBERS OF THE THESIS COMMITTEE:

[redacted]

Pah I. Chen, Chairman

[redacted]

Robert L. Stanley

[redacted]

Jack C. Riley

[redacted]

Henri B. Joyaux

This thesis presents the method of state diagram synthesis and the development of a computer program for designing fluidic sequential feedback control circuits of the fundamental mode. A paper on state diagram synthesis was authored by Chen and Lee, presented in Detroit and published as ASME paper 73-WA/Flcs-2 in 1973.

Hypothetical systems are illustrated by using series of events characterized by the piston positions of some double-acting pneumatic cylinders. In these systems, an action can only begin when the previous action has been completed. Every extension or retraction of a piston is memorized and manifested by a flip-flop element in the feedback circuit. If different combina-

tions of control signals result from different combinations of feedback signals, the logic design is straightforward. Otherwise secondary variables are needed to differentiate between repeated appearances of some ambiguous input combinations. A secondary variable is obtained as the output of a fluidic flip-flop with set and reset inputs. When a sufficient number of secondary variables are obtained, they are combined with the feedback signals. Considerations of these variables and their associated logic complementary "don't-care" conditions leads to a set of simplified control equations. The complete process of the circuit design, using state diagram synthesis, has been programmed for a digital computer.

After the control equations are obtained, one can take the signal transmission characteristics into account in order to build a hazard-free circuit.

COMPUTER AIDED DESIGN FOR FLUIDIC SEQUENTIAL CIRCUITS OF FUNDAMENTAL MODE

by

Yau-Hwang Lee

A thesis submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE
in
APPLIED SCIENCE

Portland State University
1975

TO THE OFFICE OF GRADUATE STUDIES AND RESEARCH:

The members of the Committee approve the thesis of Yau-Hwang Lee presented July 28, 1975.

[redacted]

Pah I. Chen, Chairman

[redacted]

Robert L. Stanley

[redacted]

Jack C. Riley

[redacted]

Henri B. Joyaux

APPROVED:

[redacted]

Carleton G. Fanger, Acting Head,
Department of Engineering and Applied Science

[redacted]

David T. Clark, Dean of Graduate Studies

ACKNOWLEDGEMENTS

I wish to express my appreciation to Dr. Pah I. Chen, Professors Jack C. Riley and Henri B. Joyaux for their helpful assistance in the preparation of this thesis.

I owe particular thanks to Dr. Chen, my thesis advisor, for his patient guidance, suggestions, and encouragement in the development of this thesis.

Thanks are also extended to Portland State University Computer Center which provided the computing facility needed in the thesis.

I would like to express my deepest gratitude to my parents for their support of my study in the United States of America.

TABLE OF CONTENTS

PAGE

ACKNOWLEDGEMENTS iii

LIST OF TABLES vii

LIST OF FIGURES viii

NOMENCLATURE x

CHAPTER

I INTRODUCTION 1

II STATE DIAGRAM SYNTHESIS 7

2.1 Introduction 7

2.2 Hypothetical System 7

2.3 Truth Table 8

2.4 Ambiguous State Pairs 8

2.5 Secondary Variables 11

2.6 Feedback Combinations and Redundant Variables . . 15

2.7 Techniques for Obtaining Secondary Variables . . . 17

2.8 Circuit Equations 21

2.9 External Controls and Circuit 23

2.10 Hazard-Free Circuits 25

III SOME PRELIMINARY DISCUSSION ON THE PROGRAM . . . 33

IV A SUBPROGRAM "SIMP" 35

4.1 Introduction 35

PAGE

4.2 A set of Secondary Variables 35

4.3 The Simplest Set of Secondary Variables 37

4.4 Analogy Between the Choosing of Secondary Variables and the Choosing of Output Expressions. . 38

V FEEDBACK AND CONTROL SIGNALS 41

5.1 Introduction 41

5.2 Feedback Signals 41

5.3 Control Signals 41

VI AMBIGUOUS STATE PAIRS 44

VII SET-RESET PAIRS 46

7.1 Introduction 46

7.2 The Set-Reset Pairs for the 1st-Order Secondary Variables 46

7.3 The Inapplicable Set-Reset Pairs 47

7.4 The Set-Reset Pair for the 2nd-Order Secondary Variables 48

VIII THE SIMPLEST SET OF SECONDARY VARIABLES 53

8.1 Flip-Flop Elements 53

8.2 The Simplest Set of Secondary Variables 53

IX NON-AMBIGUOUS SYSTEMS 56

X OUTPUT EXPRESSIONS 57

10.1 Introduction 57

10.2 All Possible Output Expressions 57

PAGE

10.3 The Simplest Set of Output Expressions 58

10.4 "On" and "Don't-Care" States 58

10.5 Output Equations 59

XI SUMMARY 61

BIBLIOGRAPHY 63

APPENDIX

A COMPUTER PROGRAM AND PRINTOUT 66

B FLOWCHARTS AND DATA CARDS 76

C VARIABLES CONTAINED IN THE PROGRAM 85

D DIMENSIONS OF SUBSCRIPTED VARIABLES CONTAINED IN THE PROGRAM 88

E STATE DIAGRAM, OUTPUT EQUATIONS, AND CONTROL CIRCUIT DIAGRAMS OF HYPOTHETICAL SYSTEMS 1 AND 2 90

LIST OF TABLES

TABLE		PAGE
I	A Truth Table	27
II	Elements of Array MD (I,J)	40
III	Changes of B(I)'s	40
IV	A Truth Table	43
V	The Component Signals of a Possible Set-Reset Pair	49
VI	All Possible Set-Reset Pairs	49
VII	All Possible Set-Reset Pairs Expressed in Ternary Numbers	50
VIII	All Possible Set-Reset Pairs Expressed in Decimal Numbers	51
IX	Simplified Set-Reset Pairs Expressed in Decimal Numbers	52
X	A Truth Table for a Flip-Flop Element	55

LIST OF FIGURES

FIGURE		PAGE
1	A Model of the Fundamental Mode Circuit	5
2	The Physical Arrangement of a Two-Cylinder System	6
3	The Operating Loops	27
4	The State Diagram for Secondary Variables	28
5	The State Diagram for Output Equations	29
6	An External Control Circuit	29
7	Simultaneous Change of Logic Conditions of Two Input Signals	30
8	Non-Simultaneous Change of Logic Conditions of Two Input Signals	30
9	An Ideal State Diagram of an AND Gate	30
10	A Real State Diagram of an AND Gate	30
11	An R-C Time Delay Circuit for an AND Gate	31
12	An Ideal State Diagram of an OR Gate	31
13	A Real State Diagram of an OR Gate	31
14	An R-C Time Delay Circuit for an OR Gate	31
15	A Fluidic Control Circuit for the Hypothetical Three-Cylinder System Designed by the Computer Aided Method	32
16	Simplified Flowchart for the Complete Computer Program	76
17	Flowchart for the Subprogram "SIMP"	77
18	Flowchart of Program I	78

FIGURE PAGE

19 Flowchart of Program II 79

20 Flowchart of Program III 80

21 Flowchart of Program IV 81

22 Flowchart of Program V 82

23 Flowchart of Program VI 83

24 Data Cards 84

25 State Diagram of System 1 90

26 Control Circuit Diagram of System 1 91

27 State Diagram of System 2 92

28 Control Circuit Diagram of System 2 93

NOMENCLATURE

C	Control Signals
F	Feedback Signals
r,s	State Number
R,S	Feedback Combination at States r and s
X	External Control Signals
Y	Secondary Variables
W	Auxiliary Signal for External Control

CHAPTER I

INTRODUCTION

The methods for designing a fundamental mode sequential circuit with feedback have been developed in the field of digital electronics. Several different methods have been proposed, such as the AND-OR memory circuit technique, for obtaining necessary secondary variables; the flip-flop programming for seeking the set-reset signals for some pre-assigned flip-flops; the binary counter synthesis developed according to the counting sequence of a binary counter; and the shift register method based on the nature of shift registers, Wickes (1)*, Hill and Peterson (2). The flip-flop circuits were presented by Foster and Parker (3), and later by Cheng and Foster (4), (5); the binary counter technique was proposed by Standen (6); and the shift register method by Cheng (7). To the author's knowledge, the AND-OR circuits have not been previously used in the design of fluidic control systems.

Other than the above methods, a technique involving the use of a synthesis table for seeking necessary secondary variables and circuit equations was presented by Cole and Fitch (8). Furthermore, an attempt to automatically design the control circuit with the aid of a digital computer was reported by Cheng and Foster (5). But, because of the nature of the flip-flop method which their program was based on, a program to cover the entire design process was not reported to be available.

---

*Numbers in parentheses designate References at the end of the thesis.

This thesis presents a new computer aided design technique based on the state diagram method as reported by Chen and Lee (9). The entire design process, covering all possible situations of a fundamental mode sequential feedback system, is programmed for a digital computer.

A description of the state diagram method is presented in this thesis. The steps of this method are (a) identifying ambiguous state pairs, (b) determining the secondary variables, (c) classifying the system type, and (d) detecting possible signal hazards. The objective of the thesis is to illustrate the state diagram technique for designing all fundamental mode feedback sequential circuits.

A general model of the sequential circuit with feedback is shown in Figure 1. On this figure X1, X2, ..., and Xm stand for external control signals, or the external input variables; F1, F2, ..., and Fn express the feedback signals or the feedback circuit input variables; Y1, Y2, ..., and Yp indicate the secondary variables; and C1, C2, ..., and Cp represent the circuit outputs, or the control signals. If a change in one input occurs, while there is no change in other inputs, until the transition of each secondary variable and each output signal is stabilized, the circuit is said to be operating in the fundamental mode.

In this thesis each hypothetical system is described by a series of events characterized by the positions of some pneumatic or hydraulic cylinders. Every cylinder is assumed to be double-acting. An action can only begin when the previous action has been completed. The

physical arrangement of a two-cylinder system is shown in Figure 2. The piston position limits are detected by fluidic touch sensors which are installed at the extremes of the piston strokes. Every extension or retraction of a piston is memorized and manifested by a flip-flop element in the feedback circuit.

When an operating function of a system has been selected, the logic conditions of each feedback signal and each control signal at every state is determined. A logic relationship between these two classes of signals can also be established. According to this relationship, a control circuit may be designed by using the feedback signals as inputs to obtain proper control signals for each state. However, if there are two states in the operating sequence which possess the same combination of feedback signals but result in different control signals, a secondary variable is needed to differentiate between the repeated appearance of the ambiguous feedback combination. Two such states are said to be an ambiguous state pair.

In the state diagram method, a secondary variable is obtained as the output of a fluidic flip-flop with set and reset inputs. If two states respectively preceding the two ambiguous states contain the same combination of feedback signals, one secondary may not be variable enough to distinguish this ambiguous state pair. Similarly, if several ambiguous pairs exist in series, this process can be repeated until a sufficient number of secondary variables are generated so that all ambiguous states can be differentiated.

In order to achieve the minimum number of secondary variables, all

secondary variables must be compared and reduced. Once the simplest set of secondary variables is obtained, these variables are combined with the feedback signals as the inputs for the control circuit. After the consideration of logic complementary and "don't-care" conditions, a group of simplified logic equations can be obtained. Finally, the signal transmission characteristics must be considered in order to build a hazard free circuit.

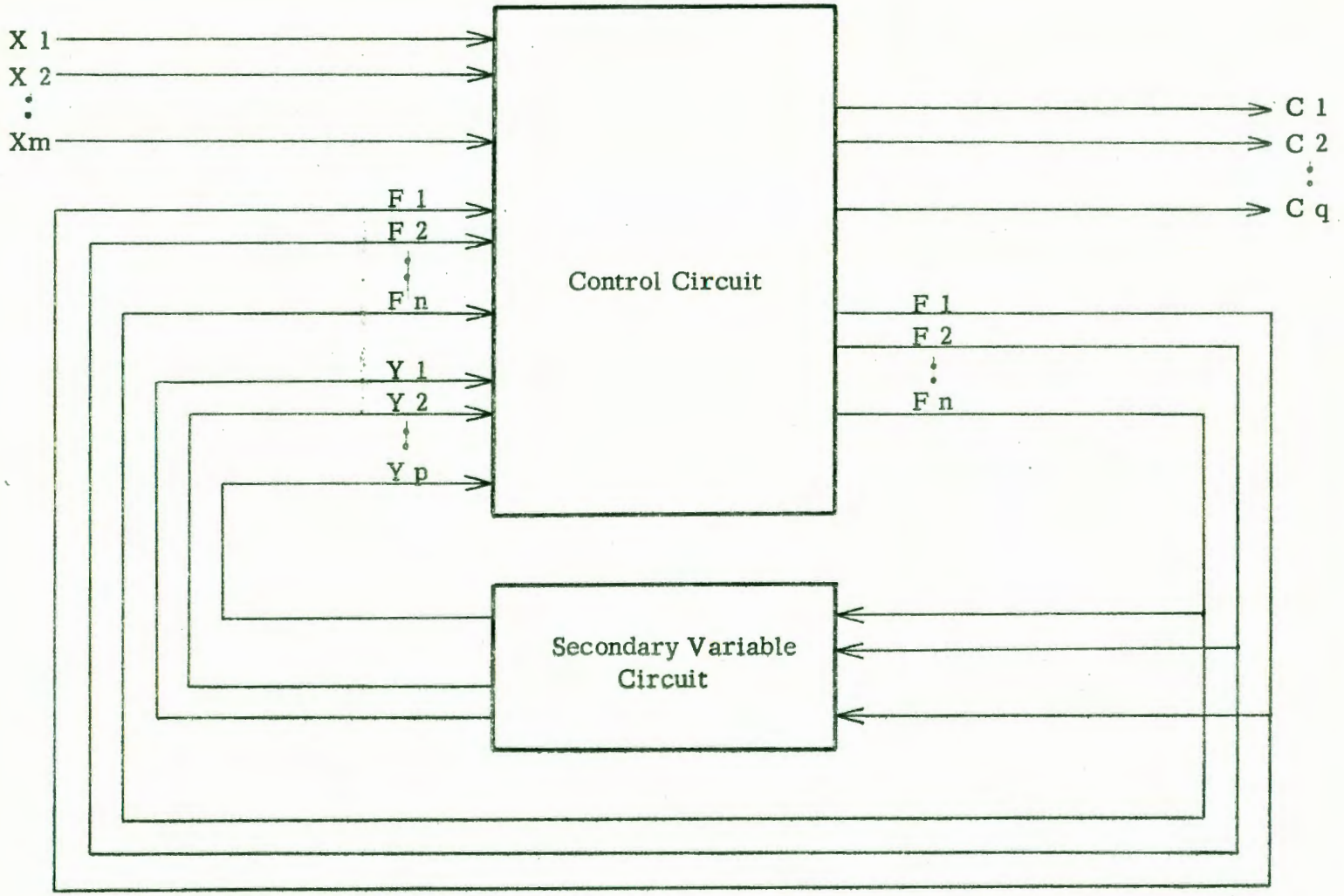


Figure 1. A model of the fundamental mode circuit

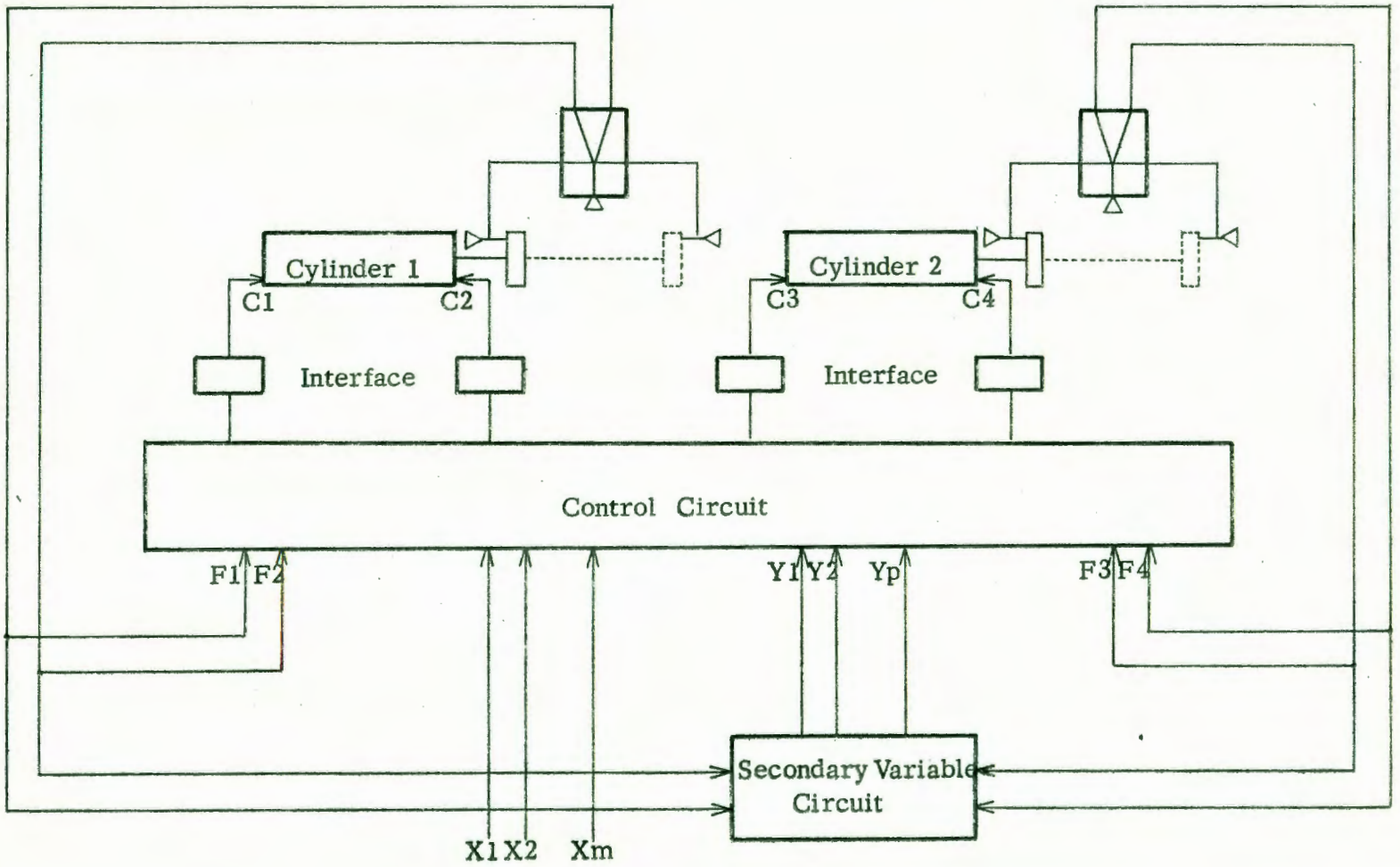


Figure 2. The physical arrangement of a two-cylinder system

CHAPTER II

STATE DIAGRAM SYNTHESIS

2.1 INTRODUCTION

The validity of a design method for the sequential control circuits depends on whether every two mutually exclusive logic combinations of output signals can be differentiated. The principle used in the state diagram method is to find a secondary variable which has different logic conditions at the two states when the output combinations are different but the input combinations are the same. The technique for finding all necessary variables for fundamental mode sequential feedback systems will be described in detail in this chapter.

2.2 HYPOTHETICAL SYSTEM

In order to illustrate the synthesis technique, assume a simple hypothetical system consisting of three pneumatic cylinders which will perform one event at each step in the following order:

Step 1 - Cylinder 1 extends

Step 2 - Cylinder 1 retracts

Step 3 - Cylinder 1 extends

Step 4 - Cylinder 2 extends

Step 5 - Cylinder 1 retracts

Step 6 - Cylinder 3 extends

Step 7 - Cylinder 3 retracts

Step 8 - Cylinder 2 retracts

Operation begins again or stop.

This system also requires a start control to initiate the operation, a cycle stop control to stop the operation and retract all cylinders at the end of the cycle, and an emergency stop control to halt the operation and retract all cylinders immediately.

2.3 TRUTH TABLE

The first step of the circuit design using the state diagram method is to indicate the logic conditions of feedback and control signals on a truth table. A truth table describes the operating function of a system by listing the logic condition of every combination of feedback and control signals together with all ambiguous pairs in the system. For the hypothetical system, the feedback signals correspond to piston positions, while the control signals are used to extend or retract their respective pistons. Once the operating function of a system has been determined, the logic conditions of these signals can be found.

The truth table of the hypothetical system is shown in Table I. On the table, "1" stands for the logic "on" condition, "0" expresses the logic "off" condition, and "ϕ" indicates the "don't-care" condition. Since only one event can occur at each step, only one control signal can be "on". These "on" controls are also shown on this table. The last column of the table contains the ambiguous state pairs which will be discussed next.

2.4 AMBIGUOUS STATE PAIRS

On the truth table, we have shown the logic conditions of all control

and feedback signals. If any two states in the system possess the same feedback signal combination but different control signal combinations, these two states are said to form an ambiguous pair of the first kind (henceforth referred to as the 1st-kind). If there are two states which immediately precede the two states of an ambiguous pair (both 1st and 2nd kinds) and possess the same feedback combination, they shall be called an ambiguous pair of the second kind (henceforth referred to as the 2nd-kind). The two states of an ambiguous pair either of the 1st or of the 2nd-kind are said to be ambiguous states.

By these definitions, we know an ambiguous pair of the 2nd-kind cannot exist independently, it must be followed by another ambiguous pair. Therefore, if some ambiguous pairs exist in series, the last pair must be of the 1st-kind. Consider a series of ambiguous pairs as a group of ambiguous pairs. The following theorem provides the nature on the existence of the ambiguous pairs.

Theorem 1. If a fundamental mode sequential feedback control system operates in cyclic order then there exists one and only one ambiguous pair of the 1st-kind in a group of ambiguous pairs. This pair must be the last pair of this group. There also exists a non-ambiguous pair which immediately precedes the first ambiguous pair of this group.

Proof: Since each system operates in a cyclic order, its operation sequence can be expressed by a loop indicated by state numbers. Two

loops express a same m-state system with an ambiguous pair of the 1st-kind at states n and r as shown in Figure 3. On this figure, these two loops are so arranged that they have states n and r in a corresponding position. Since state n and r have the same feedback signal combination but different control signal combinations, the different control signals will distinguish the same feedback combination at state n+1 and r+1. States n+1 and r+1 are then made non-ambiguous. Therefore two states cannot be ambiguous if their immediate preceding states are an ambiguous pair of the 1st-kind. On the contrary, the two immediate preceding states to an ambiguous pair cannot be an ambiguous pair of 1st-kind.

Since states n and r are ambiguous, states n-1 and r-1 cannot form an ambiguous pair of 1st-kind. If states n-1 and r-1 form an ambiguous pair of 2nd-kind, this pair can be combined together with the 1st-kind ambiguous pair at states n and r as a group. Then states n-2 and r-2 may be either a non-ambiguous pair or an ambiguous pair of the 2nd-kind. Similarly, if states n-2 and r-2 are ambiguous, states n-3 and r-3 may be either a non-ambiguous pair or an ambiguous pair of the 2nd-kind.

Since states n-1 and r-1 are non-ambiguous, if we keep checking at the two immediate preceding states to every ambiguous pair until a non-ambiguous pair appears, we will stop somewhere before or at states n+1 and r+1. Therefore, there always exists a non-ambiguous pair until a non-ambiguous pair which is immediately preceding the first ambiguous pair of a group.

The second step of the design is to find all ambiguous pairs contained in a system. According to Theorem 1, this process can be carried out by first finding the ambiguous pairs of the 1st-kind. Since the system is of the fundamental mode, only one event can occur at each state, thereby only one control signal is required to be "on" at a time. If two states which have the same feedback combination but different control combinations, then the "on" controls at these two states must be different. Therefore, to find an ambiguous pair of the 1st-kind from two states which have a same feedback combination, we simply check whether the "on" control signals at the two states are different or not. Once an ambiguous pair of the 1st-kind is found, we can check for every two states which are immediately preceding these two states of ambiguous pairs in order to find the ambiguous pair of the 2nd-kind.

When an ambiguous pair is found, it is assigned with a number and shown in the truth table (Table I). In this table, a circled number expresses an ambiguous pair of the 1st-kind, and the numbers in the same rectangular frame belong to a same group. The number 2 states are ambiguous pairs of the 2nd-kind.

2.5 SECONDARY VARIABLES

An "on" output (control) is said to be representable by the input feedback combination at this state if and only if this combination will not be repeated an any other state where this output is "off". In a system, if every

output at each "on" state is representable, this system is called a non-ambiguous system. Obviously, if there is no ambiguous pair, the system is non-ambiguous.

If an additional signal "on" at one state of an ambiguous pair and "off" at the other can be obtained, then the combination of this new signal and original inputs at these two states can be made different. So these two states are now no longer ambiguous under the introduction of this additional signal. Such an additional signal is called a secondary variable. And, the two ambiguous states are said to be differentiable between each other by this secondary variable. Apparently, if enough secondary variables are found so that every two ambiguous states of the 1st-kind are differentiable, the ambiguous system will become non-ambiguous. The following two theorems relate the ambiguous pairs and secondary variables.

Theorem 2. If the two preceding states immediate to a group of ambiguous pairs are non-ambiguous, there exists a secondary variable which can differentiate between every two ambiguous states of the first n ambiguous pairs (where $n \geq 1$).

Proof: Let state pairs s+1 and r+1, s+2 and r+2, ..., s+n and r+n, be ambiguous pairs of a same group, and also let states s and r be the non-ambiguous pair immediately preceding this group. As states s and r are non-ambiguous, their input combinations S and R must be different, so we can use S and R to set-reset a flip-flop element. Since there is a change in an input signal between every two neighboring states, the input

combinations S and S+1 at states s and s+1 are different. Because states s+1 and r+1 are ambiguous, their input combinations S+1 and R+1 are the same. This implies that the input combination S+1 is different not only from S but also from R. Therefore, the combination S+1 will not affect the flip-flop, so that the flip-flop will remain set at state s+1. Similarly, R+1 is different from both R and S, therefore the flip-flop will remain reset at state r+1. Hence the output signal of this flip-flop is "on" and "off" at states s+1 and r+1 respectively. The ambiguous states s+1 and r+1 are then differentiable by this signal. Furthermore, if no input combination at any following state in this group is the same as either S or R, the flip-flop will remain set at states s+i and reset at r+i, where i = 2,3,...,n. Then every two ambiguous states in this group are made differentiable between each other.

If, however, two ambiguous states, say s+j and r+j, have a same input combination as either S or R, the flip-flop will become set or reset at both states s+j and r+j. Since every two ambiguous states have the same input combination, from state s+j on, also from the state r+j on, the flip-flop will remain either set or reset, or change from set to reset, or from reset to set at both ambiguous states. Therefore, the output of the flip-flop can no longer be used to differentiate any ambiguous pair after the ambiguous states s+j and r+j.

Theorem 3. Any two ambiguous states of the 1st-kind can be made non-ambiguous by obtaining secondary variables through the set-reset of

flip-flop elements with pairs of feedback signal combinations together with some previously obtained secondary variables.

Proof: From Theorem 2, we know that a secondary variable may not be able to differentiate between every two ambiguous states of a group of ambiguous pairs. But since this secondary variable can differentiate the first n ambiguous pairs, it can therefore reduce the number of ambiguous pairs in this group. If we now combine this secondary variable together with the feedback signals as circuit inputs, the immediate preceding states to the reduced group may become non-ambiguous. Therefore, another secondary variable can be obtained to differentiate between the first n ambiguous pairs of the reduced group, where $n \geq 1$. Since the number of ambiguous pairs is finite, by repeating this reduction process every two ambiguous states in this group can be made non-ambiguous.

Through Theorems 1, 2, and 3, we have demonstrated that the state diagram method is able to solve any systems involving fundamental mode sequential feedback circuit. We know also from the above discussion that before the two states of an ambiguous pair of the 1st-kind are to be differentiable between each other, the two states of each ambiguous pair of the 2nd-kind in the same group must be made differentiable between each other. Therefore, although only the two states of each 1st-kind ambiguous pair are required to be differentiated between each other, we must be able not only to differentiate between the two states of an ambiguous state pair of the 1st-kind but also the 2nd-kind in the system.

In order to clearly classify the systems let us define the secondary variables which are obtained by only using the combinations of feedback signals as set-reset signals as first order secondary variables. These are obtained by using the combinations of feedback signals with some other secondary variables as set-reset pairs as second order secondary variables. Under this classification, a system requiring no secondary variable shall be called a first type system. A system containing only second kind secondary variables shall be called a second type system. A system that requires both first and second order secondary variables shall be called a third type system (henceforth 1st-order, 2nd-order, 1st-type, 2nd-type, and 3rd-type will be used instead of first order, second order, first type, second type, and third type).

2.6 FEEDBACK COMBINATIONS AND REDUNDANT VARIABLES

We have thus far used the words "feedback signal combination at a state" several times. This terminology represents the combination of feedback signals at a certain state. From now on, we will also use another expression "a possible feedback signal combination". For example, the feedback signal combination at state 1 in our hypothetical system is $F1\overline{F2}F3\overline{F4}F5\overline{F6}$. But if $\overline{F2}$=F1, $\overline{F4}$=F3, and $\overline{F6}$=F5, this combination can then be expressed by F1F3F5. A possible feedback combination at state 1 can be any of the combinations: F1, F3, F5, F1F3, . . . , or F1F3F5. Since only three of the six feedback signals are independent, all possible feedback

signal combinations in this system are F1, F2,..., F6, F1F3,..., F4F5,..., F1F3F5,..., and F2F4F6. If a secondary variable Y1 can differentiate between every two ambiguous states that can be done by another secondary variable Y2, but not vice versa, then Y2 is a redundant secondary variable with respect to Y1. However, if Y2 can also perform the function of Y1, let us assign S1 and S2 as the number of logic elements which are required to generate Y1 and Y2 respectively; when S1< S2, then Y2 is a redundant secondary variable, when S1=S2, then either Y1 or Y2 can be redundant.

The number of logic elements required for generating a secondary variable depends on the type of elements being selected. For instance, if we use F1F3F5 and F2F3F6 as a set-reset pair, we may need four 2-input AND elements to obtain the set-reset signals with (F1F3)F5 as set and (F2F3)F6 as reset, or we need two 3-input NOR gates by using $\overline{\overline{F1}+\overline{F3}+\overline{F5}}$

as set and $\overline{\overline{F2}+\overline{F3}+\overline{F6}}$ as reset. In order to make the numbers of required logic elements comparable, we assume that the number of logic elements required for a secondary variable is proportional to the number of signals presented in the set-reset pair. For example, the number of signals in the set-reset pairs F1F3F5-F2F3F5 is 6 while in F1F3-F2F4F6 it is 5, therefore we say the former secondary variable needs more logic elements than the latter. Each feedback or secondary signal in a set-reset pair is called a component signal of this pair.

2.7 TECHNIQUES FOR OBTAINING SECONDARY VARIABLES

If no ambiguous state pair exists in a system, the state diagram consisting of feedback and control signals can be formed directly to obtain the control equations. On the other hand, if there is an ambiguous pair, there is a secondary variable. Several different techniques are available for obtaining sufficient secondary variables according to different selections of set-reset pairs.

(1) Fast Design.

This technique uses the feedback signal combinations at the two immediate previous states of a group of ambiguous pairs to set-reset a flip-flop element. It has been proven in Section 2.5 that this technique can provide a sufficient number of secondary variables.

In order to make a comparison among secondary variables so that redundant secondary variables can be removed, the logic condition of

every variable at each state must be shown clearly. So we find secondary variables with the aid of a state diagram as shown in Figure 4.a. In this figure, the first row shows the index number of each state; the second row exhibits the feedback combination at each state (135 represents F1F3F5); from the third row on, each row presents the information about the indicated secondary variable.

The process of obtaining each secondary variable begins with the indexing of the feedback signals which will be used as a set-reset pair below the words "SET" and "RESET". With these signals, we can set-reset a flip-flop element through a complete cycle. Let us indicate the logic "on" or "off" condition of this variable at a state with a high or low level segment then set aside the index number of these ambiguous pairs whose two states can be differentiated between each other by this variable. Repeating this process for each group, we can find sufficient secondary variables of the 1st-order.

Once every secondary variable for each ambiguous group is found, we can eliminate redundant variables among them according to the definition of redundant variables.

After that, each necessary variable is assigned an index number. Secondary variables are designated by odd index numbers similar to those of feedback signals. Therefore, on the state diagram, only Y1,Y3, ..., appear while Y2,Y4..., will be used to indicate the inverse of Y1,Y3, ..., respectively. If some ambiguous groups still exist, we shall combine

the feedback signals at the two immediately previous states to the group together with a secondary variable which has different logic conditions at these two states as a set-reset pair to get another secondary variable. Then, eliminate the redundant variables among the new secondary variables, and repeat this process until sufficient secondary variables are obtained so that every two ambiguous states in this system can be made differentiable between each other.

(2) State Diagram Method

As discussed in the above example, the feedback combinations at states 2 and 7 can be used to set-reset a flip-flop element so that a secondary variable can be obtained in order to differentiate state 1 from state 3. However, these two combinations are not the only pair known to perform this function. This can be seen if we assign the logic conditions of a secondary variable at states 1 and 3 as "on" and "off" respectively by high and low level solid segments as shown in Figure 4.b, and choose a pair of feedback combinations at two different states to set-reset a flip-flop. We can obtain three different pairs of combinations that can be used to form a secondary variable for generating the desired logic conditions at states 1 and 3. Because of the bistable characteristic of the flip-flop elements, if we assign the logic condition of a secondary variable at state 1 as "off" and 3 as "on" there is no additional pair of feedback combinations other than the above three that can be found to obtain a

secondary variable in order to meet the required logic conditions. Therefore, these three secondary variables are all that we can find. Their logic conditions are shown with dotted segments in Figure 4.b. Similarly, we can find three different pairs of feedback combinations to obtain a secondary variable in order to differentiate between the ambiguous states 6 and 8.

But it is not impossible to find any set-reset pair for the ambiguous states 2 and 4. Therefore, we know a 2nd-order secondary variable is required for this ambiguous pair. This system is then of the 3rd-type.

Next we shall eliminate all redundant variables among these six 1st-order secondary variables. We find that the secondary variable with F1F3F5-F2F4F6 as set-reset pair is all that we need for both ambiguous pairs of states 1 and 3 and states 6 and 8. Considering all previously obtained but not eliminated secondary variables together with feedback signals as input signals and repeating the same process as outlined above for obtaining the 1st-order secondary variables, we can find all necessary 2nd-order secondary variables as shown in Figure 4.b.

Comparing this process with the previous method, we can see that this method may be a little more time consuming but results in fewer secondary variables.

(3) Computer Aided Design

Besides the feedback combinations at eight operating states, there still exist many possible feedback combinations. This means that there are some more set-reset pairs available in this system. Figure 4.c

shows two simple set-reset pairs with which every two ambiguous states can be differentiated between each other. But generally, it takes extra design time to find such simple set-reset pairs. So the method using all possible feedback combinations as set-reset pairs is used only in the computer aided design to be discussed in Chapter 6.

In order to achieve the proper function of each secondary variable, the initial condition of every flip-flop element must be considered. Therefore, if a flip-flop element is not activated at the initial state, then it must be initialized by the system start signal. For example, the set of Y1 and reset of Y3 in the fast design method, the set of Y1 in the state diagram method, and the set of Y1 and reset of Y3 in the computer aided method must be initialized.

2.8 CIRCUIT EQUATIONS

Once all necessary secondary variables are obtained, they can be combined with the feedback signals and control signals to form a state diagram as shown in Figure 5. Since $\overline{F2}$=F1, $\overline{F4}$=F3, and $\overline{F6}$=F5, only F1, F3, and F5 are needed on the diagram. Let us use solid segments on the diagram to represent logic "on" or "off" conditions and dotted segments to stand for "don't-care" conditions.

We shall now proceed to find the control equations by examining the state diagram. By observation, we can find the simplest input combination at state 1 to represent the control signal C1 which is F1F3. Similarly, the

simplest combinations for C1 at states 3 and 4 are F1F3 and F3Y3 respectively. If we underline a combination which represents the "don't-care" states, the logic equation of C1 can be expressed as:

$C1=F1F3+\underline{F3Y3}$

Similarly, we can get the following:

$C2=\overline{F1}\,\overline{Y3}+\overline{F3}$

$C3=\overline{F1}Y3+\underline{\overline{F3}\,\overline{Y1}}+\underline{\overline{F5}}$

$C4=\underline{F1F3}+F5\overline{Y3}$

$C5=F1\overline{F3}\,\overline{Y1}$

$C6=\underline{F3}+\underline{Y1}+\overline{F1}$

From the state diagram of Figure 5, we can see that if each "don't-care" condition of every control signal is considered as an "on" condition, then C1 and C2, C3 and C4, and C5 and C6 become complementary pairs. Therefore, the control circuit for C1 and C2 can be built in several ways:

$C1=F1F3$

$C2=F1\overline{Y3}+\overline{F3}$

or $C1=F1F3+F3Y3$

$C2=\overline{C1}$

or $C1=\overline{C2}$

$C2=F1\overline{Y3}+\overline{F3}$

Apparently, the third set of equations is the simplest and therefore can be adopted together with the following selection of C3, C4, C5, and

C6 to build the control circuit.

$C3=\overline{F1}Y3$

$C4=F5\overline{Y3}$

$C5=\overline{C6}$

$C6=F3+\overline{F1}+Y1$

Note that simpler solutions for C3 and C4 can be found by not observing the above restriction.

2.9 EXTERNAL CONTROLS AND CIRCUIT

In this system, a start control is required to initiate the operation, an emergency stop control is needed to halt the operation and retract all cylinders immediately, and a cycle stop control is required to stop the operation at the end of the cycle when the cylinders are all retracted. Since the operation begins with the extension of cylinder 1, the function of the start control can be achieved if we restrict the extension of cylinder 1 to the condition that the control signal C1 and the start control X1 are both "on". Analyzing the function of the emergency stop control, we can assume that the emergency stop signal X2 will turn on the control signals C2, C4, and C6, and turn off the control signals C1, C3, C5. As to the cycle stop control, let us consider that the appearance of this signal will turn off the start signal X1. Further, we must find an auxiliary signal W in the system which is "off" at the initial state and "on" at

all other states where C1 is "on". Then, the cycle stop control can be achieved by extending cylinder 1 in the situation that C1 AND (W OR X1) is "on". That is, when the cycle stop is applied, the extension of cylinder 1 at the initial state is prevented because both signals W and X1 are "off". But at other states cylinder 1 can still be extended because of the existence of signal W. The arrangement of these external controls according to the above discussion is shown in Figure 6.

In the hypothetical system, the secondary variable Y1 can be used as the signal W. If the signal W cannot be found, we have to generate one as we did for secondary variables. Since the requirements for external controls are different for different systems, the design of these controls will not be included in the computer program in order to maintain its generality.

With the addition of the external controls, the final equations become

$C=\overline{C2}\,(\overline{Y1}+X1)$

$C2=F1\overline{Y3}+\overline{F3}+X2$

$C3=\overline{F1}Y3\overline{X2} = \overline{F1+\overline{Y3}+Y2}$

$C4=F5\overline{Y3}+X2$

$C5=\overline{C6}$

$C6=\overline{F1}+F3+\overline{Y1}+X2$

2.10 HAZARD FREE CIRCUITS

The signal transmission is a function of fluid impedance which in turn is a function of the area and shape of the tubing cross section, and the length and material of the transmission line. Without carefully considering the transmission impedance, a circuit may contain signal hazards. The detection of hazards in circuits is not an easy task. However, possible signal hazards in circuits designed according to the state diagram method can be located and removed.

Consider a system which contains two inputs I1 and I2 whose logic conditions change between the two same two states, say 3 and 4, as shown in Figure 7. Because the transmission conditions of I1 and I2, such as source pressure, fluid, impedance, etc., can hardly be the same, the probability for I1 and I2 to change simultaneously is almost nil. That is, the change of I1 will either lead or lag the change of I2 as shown in Figure 8. This non-simultaneous change of I1 and I2 becomes the source of a signal hazard. According to the components selected, the signal hazards can be classified into two kinds: one is produced by employing AND elements, and the other by OR gates.

(1) Hazards in AND Gates

Consider a state diagram (Figure 9) in which the output signal O1 is I1I2. When the change of I1 leads that of I2, a signal hazard between states 3 and 4 occurs as indicated by Figure 10. One way to remove this hazard is to delay the transmission of I1 so that the change of I1

is always lagging behind the change of I2. In Figure 11, an R-C circuit is used to delay the arrival of I1.

(2) Hazards in OR Gates

Let us consider another state diagram (Figure 12) where O1=I1+I2. If the change of I1 leads that of I2, a hazard between states 3 and 4 will be produced as shown in Figure 13. The same R-C circuit to delay the input signal I1 as used before can remove this hazard as shown in Figure 14.

From the above discussions, we know that signal hazards can be produced when some input signals to a logic gate change their logic conditions with different time delays. These hazards can be removed by using the R-C circuits.

Let us check the state diagram of the hypothetical system in Figure 5. We notice that the logic conditions of F1 and Y3 change between states 2 and 3, so a pulse signal may be created by the combination $\overline{F1}Y3$ in the C3 circuit. Therefore, Y3 must be delayed. Similarly, an R-C circuit is added to the C4 circuit to prevent a possible hazard from happening by using a combination of F5 and Y3 with an AND gate between states 6 and 7. The hazard free circuit is shown in Figure 15. The subcircuit shown in the dotted enclosure represents the main circuit which generates the control signals. The subcircuit above the enclosure represents the external control, while that on the left is used to generate necessary secondary variables.

TABLE I

A TRUTH TABLE

State	F1	F2	F3	F4	F5	F6	C1	C2	C3	C4	C5	C6	Control	Amb. pr.
1	1	0	1	0	1	0	1	0	0	ø	0	ø	C1	2
2	0	1	1	0	1	0	0	1	0	ø	0	ø	C2	①
3	1	0	1	0	1	0	1	0	0	ø	0	ø	C1	2
4	0	1	1	0	1	0	ø	0	1	0	0	ø	C3	①
5	0	1	0	1	1	0	0	1	ø	0	0	ø	C2	
6	1	0	0	1	1	0	0	ø	ø	0	1	0	C5	③
7	1	0	0	1	0	1	0	ø	ø	0	0	1	C6	
8	1	0	0	1	1	0	0	ø	0	1	0	ø	C4	③

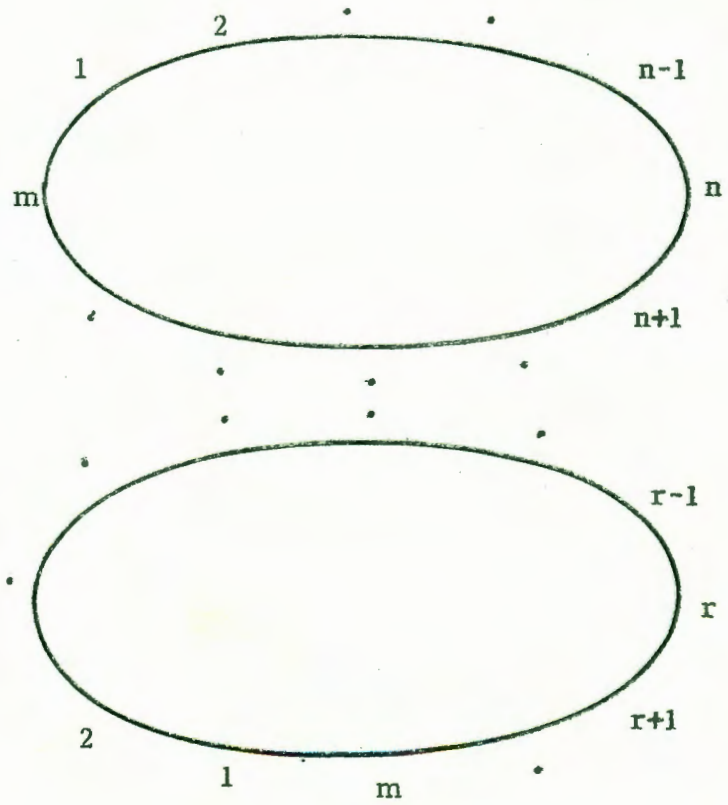
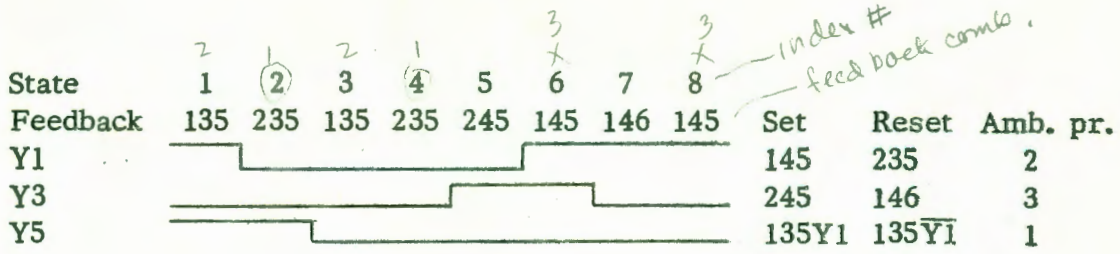


Figure 3. The operating loops



(a) The fast design method

State	1	2	3	4	5	6	7	8			
Feedback	135	235	135	235	245	145	146	145	Set	Reset	Amb. pr.
									145	235	2
Y1									146	235	2, 3
									245	235	2
									245	146	3
									235	146	2, 3
									135	146	3
Y3									135Y1	135$\overline{Y1}$	1

(b) The state diagram method

State	1	2	3	4	5	6	7	8			
Feedback	135	235	135	235	245	145	146	145	Set	Reset	Amb. pr.
Y1									6	2	2, 3
Y3									1$\overline{Y1}$	6	1

(c) The computer aided method

Figure 4. The state diagram for secondary variables

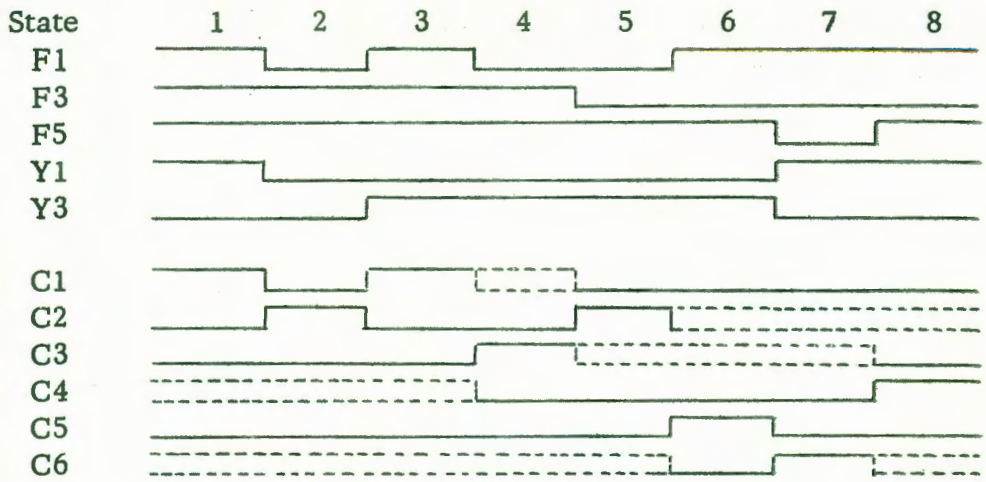


Figure 5. The state diagram for output equations

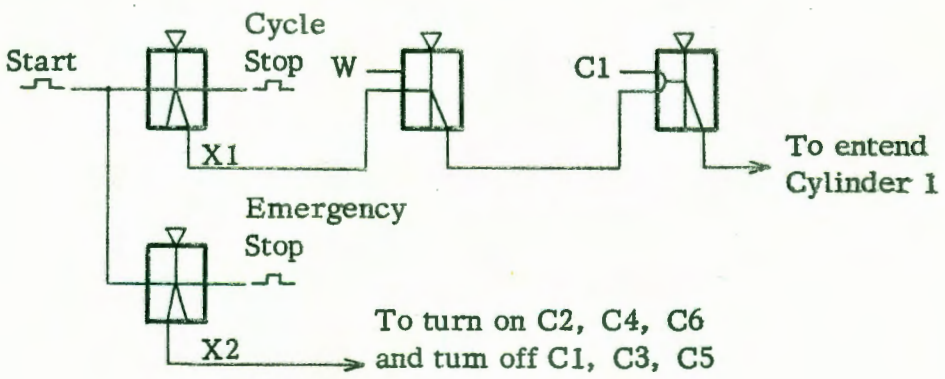


Figure 6. An external control circuit

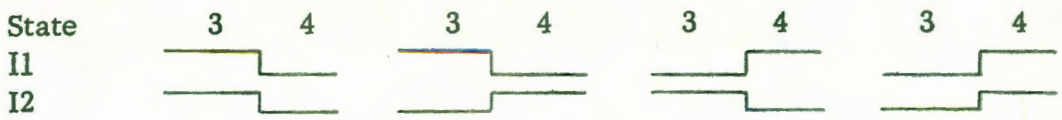


Figure 7. Simultaneous change of logic conditions of two input signals

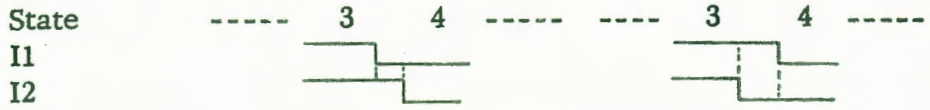


Figure 8. Non-simultaneous change of logic conditions of two input signals

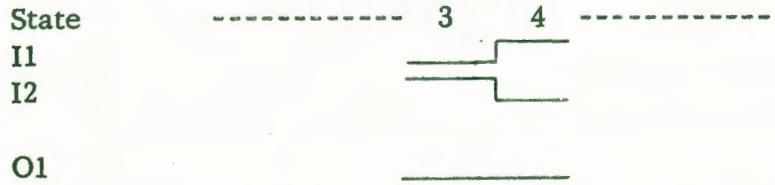


Figure 9. An ideal state diagram of an AND gate

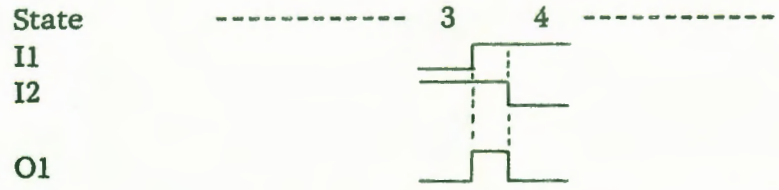


Figure 10. A real state diagram of an AND gate

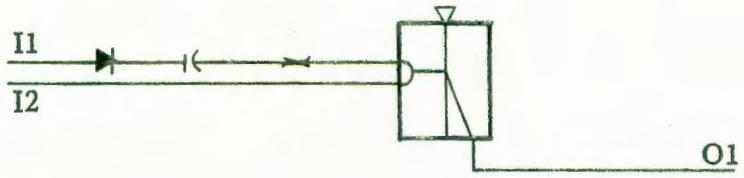


Figure 11. An R-C time delay circuit for an AND gate

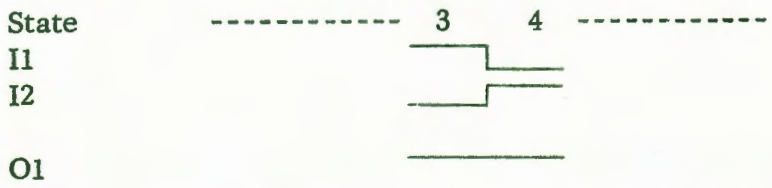


Figure 12. An ideal state diagram of an OR gate

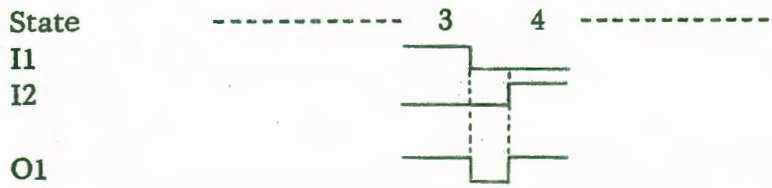


Figure 13. A real state diagram of an OR gate

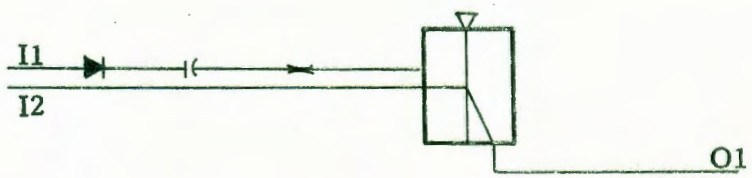


Figure 14. An R-C time delay circuit for an OR gate

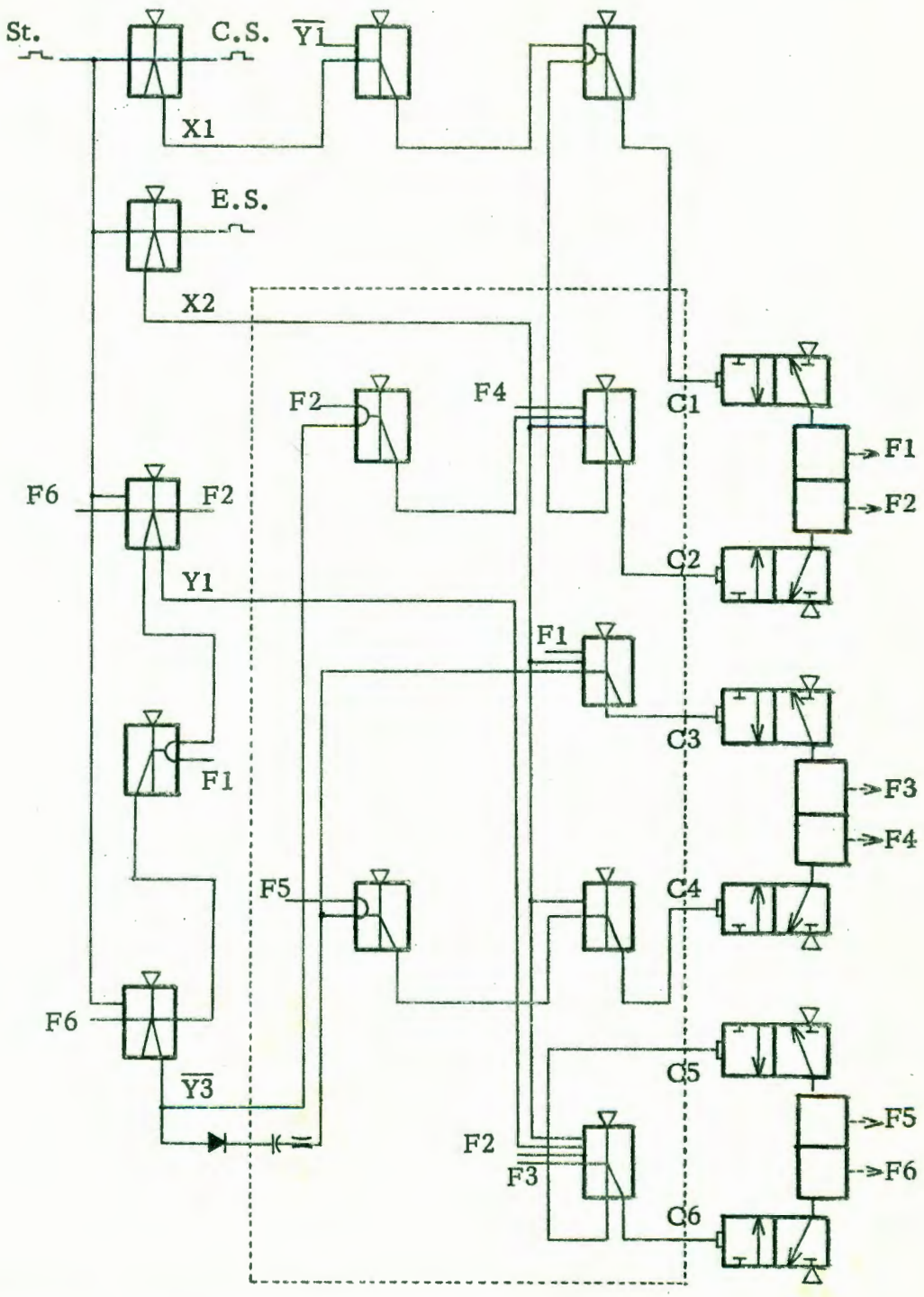


Figure 15. A fluidic control circuit for the hypothetical three-cylinder system designed by the computer aided method

CHAPTER III

SOME PRELIMINARY DISCUSSION ON THE PROGRAM

The computer program for designing sequential control circuits presented in this thesis is developed by using FORTRAN IV language on an IBM 1130. The complete program with printout is shown in Appendix A while flowcharts and data cards are shown in Appendix B.

The meaning of variables contained in the program and the dimensions of the subscripted variables in the program are listed in Appendixes C and D. All variables except the alphameric strings are defined in the integer field.

Data required in this program are: (1) seven alphameric strings "F", "C", "Y", "ON", "DONT", "SOME", and "NO" which are used respectively to express the feedback signals, the control signals, the secondary variables, the logic "on" conditions, the "don't-care" conditions and some or no ambiguous state(s) in the system; (2) the numbers of cylinders and operating states in the system; (3) the logic conditions of feedback signals as discussed before. Since a bistable condition exists at each cylinder, only F1, F3, F5,..., are needed. Also required is (4) a blank card to indicate the end of data.

In Figure 24, data cards for three systems of different types are illustrated. The first card is used for the seven alphameric strings - each string occupies four columns. The second card indicates the number of cylinders and the number of operating states of the first system in

logic conditions of feedback signals each of which consists of one feedback signal with two columns per state. For the second system, the next card will express the number of cylinders and the number of operating states of the second system, and then the logic conditions of independent feedback signals. Similarly, the following cards provide the same data for the third system. The last card, a blank card, will give an "END" command to the computer operation system. State diagrams, logic equations, and control circuit diagrams of system 1 and 2 are shown in Appendix E while that of system 3 are contained in Chapter 2.

CHAPTER IV

A SUBPROGRAM - "SIMP"

4.1 INTRODUCTION

Let us consider a system which contains NN ambiguous pairs and NP secondary variables. If an array MD (I,J) is formed such that the row elements in the Ith row are the index numbers of the secondary variables that can be used to differentiate the two states of the Ith ambiguous pair, then a set of secondary variables can be obtained by taking a secondary variable for each row of the array MD, thus making all the NN ambiguous pairs differentiable.

A program which can select the simplest set is developed in this chapter.

4.2 A SET OF SECONDARY VARIABLES

Let A(I) be the number of total row elements in the Ith row of the array MD, (I,J) as shown in Table II. From this table we know we are able to define all possible sets of MD(1,B(1)), MD(2,B(2)),..., MD(NN,B(NN)) in terms of B(I)'s in the DO loops.

```
DO 100 B(1)=1,A(1)
DO 100 B(2)=1,A(2)
.....
DO 100 B(NN)=1,A(NN)
----------
```

```
----------------------

      100   CONTINUE
```

There are NN DO statements in this program each of which is written for an ambiguous pair. If the number of ambiguous pairs changes, the number of DO statements will change accordingly. Therefore, before using this program we must know the number of ambiguous pairs. However, it is impossible to know this number in an unsolved system. So further development to cover this aspect in the program becomes desirable.

If the above program is used to solve a system with NN=3, A(1)=2, A(2)=2, A(3)=3, the combinations of B(1), B(2), and B(3) can be listed as shown in Table III. From this table we can develop a rule for the change of B(1), B(2) and B(3) from one loop to another.

(1) IF B(I)<A(I), then B(I) changes to B(I)+1, and GOTO (3)

    IF B(I)=A(I), then B(I) changes to 1, and GOTO (2)

(2) IF I>1, then I changes to I-1, and GOTO (1)

    IF I=1, then GOTO (4)

(3) Change is performed

(4) All loops have been executed, stop.

With this rule, if we have B(1), B(2), and B(3) in a loop, we can find out B(1), B(2), and B(3) in the next loop. Therefore, start with B(1)=B(2)=B(3)=1 and apply this rule repeatedly to obtain all possible combinations of the B(I)'s. This process can be programmed as follows:

```
    DO 100 I=1,NN
100 B(I)=1
--------------------------
110 ----------------------
--------------------------
    I=NN
190 IF(B(I)-A(I))200,210,200
200 B(I)=B(I)+1
    GOTO 110
210 B(I)=1
    I=I-1
    IF(I)190,220,190
220 END
```

This is a general program developed for any number of ambiguous pairs.

4.3 THE SIMPLEST SET OF SECONDARY VARIABLES

According to the definition of the redundant secondary variables, the simplest set consists of a minimum number of variables. If two sets have the same number of variables, the one containing a larger amount of component signals in the set-reset pairs is redundant.

Consider a set of secondary variables whose index numbers are MD(1,B(1)), MD(2,B(2)),...,MD(NN,B(NN)). If the secondary

variable with index number MD(1,B(1)) is different from any other secondary variables in this set, then MD(1,B(1)) must be different from any other MD(I,B(I)). That is, all the differences MD(1,B(1))-MD(2,B(2)), MD(1,B(1))-MD(3,B(3)),...,MD(1,B(1))-MD(NN,B(NN)) must not be equal to zero. Therefore, the number of secondary variables in a set can be found by applying this rule.

A variable E(I) is used in the program to represent the number of component signals for a secondary variable with index number I, so we can compare the amount of component signals between two sets of variables when these two sets contain the same number of independent variables.

4.4 ANALOGY BETWEEN THE CHOOSING OF SECONDARY VARIABLES AND THE CHOOSING OF OUTPUT EXPRESSIONS

If every "on" or "don't-care" state of a control signal in an operating cycle is representable by some input expression, we can assign different expressions with different index numbers, and form an array MD(I,J) in a manner such that elements in the Ith row of this array are the index numbers of the expressions which can represent the signal at the Ith state. Then there is an analogy between obtaining the simplest set of output expressions to represent a control signal at every "on" and "don't-care" state versus the obtaining of the simplest combination of the secondary variables to differentiate between every two ambiguous states.

Therefore, the above program is written as a subprogram that can be called upon for either case.

TABLE II

ELEMENTS OF ARRAY MD(I,J)

Ambiguous pair	Reference number of secondary variable
1	MD(1, 1), MD(1, 2), ..., MD(1, A(1))
2	MD(2, 1), MD(2, 2), ..., MD(2, A(2))
.	.....
.	.....
NN	MD(NN, 1), MD(NN, 2), .., MD(NN, A(NN))

TABLE III

CHANGES OF B(I)'s

Loop	B(1)	B(2)	B(3)	Change
1	1	1	1	
2	1	1	2	I3: 1-2.
3	1	1	3	I3: 2-3.
4	1	2	1	I3: 3-1; I2: 1-2.
5	1	2	2	I3: 1-2.
6	1	2	3	I3: 2-3.
7	2	1	1	I3: 3-2; I2: 2-1; I1: 1-2.
8	2	1	2	I3: 1-2.
9	2	1	3	I3: 2-3.
10	2	2	1	I3: 3-1; I2: 1-2.
11	2	2	2	I3: 1-2.
12	2	2	3	I3: 2-3.

CHAPTER V

FEEDBACK AND CONTROL SIGNALS

5.1 INTRODUCTION

Although the necessary data needed in the state diagram method are the logic conditions of each feedback and control signal, the existence of some logic relationships between the feedback signals and control signals requires only the logic conditions of the independent feedback signals to enter as inputs to the program. These logic relationships are presented in this chapter.

5.2 FEEDBACK SIGNALS

Let F(I) and F(J) be two feedback signals associated with the same cylinder and let F(I,K) and F(J,K) be the logic conditions of F(I) and F(J) at state K, then the complementary condition between F(I) and F(J) can be written as:

$$F(I,K)=1-F(J,K)$$

Therefore, if the logic conditions of F(1), F(3),...,F(2*NC-1), are given, where NC is the number of cylinders in this system, then the logic conditions of F(2), F(4),..., F(2*NC), can be obtained.

5.3 CONTROL SIGNALS

If we define the logic conditions of the control signals in the integer field, and let "0", "1", and "2" express the logic conditions "off",

"don't-care", and "on" respectively, then the truth table (Table I) of the hypothetical system can be redone as Table IV. From this table, logic relationships between the feedback and control signals can be found as follows:

(1) If F(I,J)=1, F(I,K)=0, then C(I,J)=2

(2) If F(I,J)=1, F(I,K)=1, then C(I,J)=0

(3) If F(I,J)=0, F(I,K)=1, then C(I,J)=0

(4) If F(I,J)=0, F(I,K)=0, then C(I,J)=1

where state K is next to state J. These relationships can be formulated as the following equation:

C(I,J)=(F(I,J)+F(I,K)+1)*(1-F(I,K))

From this equation, if the logic conditions of the feedback signals are known, the logic conditions of the control signals can be found.

TABLE IV

A TRUTH TABLE

State	F1	F2	F3	F4	F5	F6	C1	C2	C3	C4	C5	C6
1	1	0	1	0	1	0	2	0	0	1	0	1
2	0	1	1	0	1	0	0	2	0	1	0	1
3	1	0	1	0	1	0	2	0	0	1	0	1
4	0	1	1	0	1	0	1	0	2	0	0	1
5	0	1	0	1	1	0	0	2	1	0	0	1
6	1	0	0	1	1	0	0	1	1	0	2	0
7	1	0	0	1	0	1	0	1	1	0	0	2
8	1	0	0	1	1	0	0	1	0	2	0	1

CHAPTER VI

AMBIGUOUS STATE PAIRS

In Chapter 2, ambiguous pairs have been classified into two kinds. This classification simply makes it easy to find the ambiguous pairs. After each ambiguous pair has been located, all we have to do is to find a secondary variable to differentiate the two ambiguous states from each other. Therefore, it is no longer necessary to distinguish these two kinds of ambiguous pairs.

Let NF=2*NC be the total number of the feedback signals (also, the number of control signals). From the truth table (Table IV), if states I and J have the same logic combination of feedback signals, then

$$\sum_{K=1}^{NF} \left| F(K,I) - F(K,J) \right| = 0$$

Also, if they have different logic combinations of control signals then

$$\sum_{K=1}^{NF} \left| C(K,I) - C(K,J) \right| \neq 0$$

Therefore, the definitions of ambiguous pairs can be written as follows:

(1) If states I and J are ambiguous states of the 1st-kind, then

$$\sum_{K=1}^{NF} \left| F(K,I) - F(K,J) \right| = 0$$

and

$$\sum_{K=1}^{NF} \left| C(K,I) - C(K,J) \right| \neq 0$$

(2) If states I and J are ambiguous states of the 2nd-kind, then

$$\sum_{K=1}^{NF} \left| F(K,I) - F(K,J) \right| = 0$$

and

$$\sum_{K=1}^{NF} \left| F(K,M) - F(K,N) \right| = 0$$

where states M and N are the next states to states I and J respectively.

With the aid of these definitions, a program to obtain the ambiguous pairs can be developed.

CHAPTER VII

SET-RESET PAIRS

7.1 INTRODUCTION

A secondary variable can be obtained by the set-reset of a flip-flop element with a pair of input combinations. This chapter describes a computer program for finding all possible set-reset pairs.

7.2 THE SET-RESET PAIRS FOR THE 1ST-ORDER SECONDARY VARIABLES

A set or reset signal for a 1st-order secondary variable is composed of certain feedback signals. There are six feedback signals in a three-cylinder system. By considering the bistable conditions, only three signals out of the six are independent. Therefore, let F(0) express a pseudo-signal, i.e., "nothing". Therefore, a set or reset signal can consist of three feedback signals. The first feedback signal is either F(1) or F(2) or F(0), while the second feedback signal is either F(3) or F(4) or F(0), and the third feedback signal is either F(5) or F(6) or F(0). This is shown in Table V.

Let us use MC(1,1), MC(1,2), MC(1,3) and MC(2,1), MC(2,2), MC(2,3) to express the index numbers of the first, second and third component signals in a set-reset pair. Then all possible set-reset pairs can be shown in Table VI.

Since a component signal in a set or reset signal has three possible

selections, we may consider each component signal as a digit in the ternary system. Also by treating a set or reset signal as a ternary number, we can re-formulate Table VI to Table VII. Moreover, if we transfer these ternary numbers to the decimal field, we can obtain Table VIII.

The "0" in Table VIII means "0 0 0" in Table VI. It indicates the feedback combination F(0)*F(0)*F(0). Since F(0) means "nothing", so does the combination F(0)*F(0)*F(0). Therefore, any set-reset pair containing this combination as a set or reset signal is "meaningless". Besides, if a set-reset pair is composed of the same signal for use as both set and reset, such as (1-1), (2-2)...(26-26), then the output signal of the flip-flop element becomes unstable. Furthermore, if we use (1-2) to set-reset a flip-flop, the output signal of another flip-flop set-reset by (2-1) is redundant. By removing all the above set-reset pairs, we can simplify Table VIII to Table IX.

Since a decimal number can be transferred to the ternary field, we can convert a pair of decimal numbers to their corresponding feedback combinations. Therefore, all possible set-reset pairs can be obtained by finding their corresponding decimal number as shown on Table IX.

7.3 THE INAPPLICABLE SET-RESET PAIRS

When the logic conditions of a pair of set and reset signals are

both "on" at a certain state, the output signal of the associated flip-flop becomes undefined. This type of set-reset pairs shall be removed.

Let T(1,k)-T(2,k) be the logic conditions of a set-reset pair at state K. If both set and reset signals are "on" at this state, then T(1,k)*T(2,k) is equal to 1, otherwise it is equal to 0. Therefore, the inapplicable set-reset pairs can be detected and removed.

7.4 THE SET-RESET PAIRS FOR THE 2ND-ORDER SECONDARY VARIABLES

The process for finding the 2nd-order secondary variables is the same as that used for finding the 1st-order secondary variables when we consider all previously obtained but not removed secondary variables together with feedback signals as circuit inputs as was discussed in Section 2.7. Therefore the above discussion is valid for both 1st-order and 2nd-order variables.

TABLE V

THE COMPONENT SIGNALS OF A POSSIBLE SET-RESET PAIR

	SET	RESET
Signal 1	F(0), or F(1), or F(2)	F(0), or F(1), or F(2)
Signal 2	F(0), or F(3), or F(4)	F(0), or F(3), or F(4)
Signal 3	F(0), or F(5), or F(6)	F(0), or F(5), or F(6)

TABLE VI

ALL POSSIBLE SET-RESET PAIRS

SET			RESET		
Signal 1	Signal 2	Signal 3	Signal 1	Signal 2	Signal 3
MC(1, 1)	MC(1, 2)	MC(1, 3)	MC(2, 1)	MC(2, 2)	MC(2, 3)
0	0	0	0	0	0
0	0	0	0	0	5
0	0	0	0	0	6
0	0	0	0	3	0
0	0	0	0	3	5
0	0	0	0	3	6
.	.	.	.	.	.
.	.	.	.	.	.
0	0	0	2	4	6
0	0	5	0	0	0
0	0	5	0	0	5
0	0	5	0	0	6
.	.	.	.	.	.
.	.	.	.	.	.
0	0	5	2	4	6
.	.	.	.	.	.
.	.	.	.	.	.
2	4	6	0	0	0
2	4	6	0	0	5
2	4	6	0	0	6
.	.	.	.	.	.
.	.	.	.	.	.
2	4	6	2	4	6

TABLE VII

ALL POSSIBLE SET-RESET PAIRS
EXPRESSED IN TERNARY NUMBERS

SET			RESET		
Signal 1	Signal 2	Signal 3	Signal 1	Signal 2	Signal 3
MC(1, 1)	MC(1, 2)	MC(1, 3)	MC(2, 1)	MC(2, 2)	MC(2, 3)
0	0	0	0	0	0
0	0	0	0	0	1
0	0	0	0	0	2
0	0	0	0	1	0
0	0	0	0	1	1
0	0	0	0	1	2
0	0	0	0	2	0
.	.	.	.	.	.
.	.	.	.	.	.
0	0	0	2	2	2
0	0	1	0	0	0
0	0	1	0	0	1
0	0	1	0	0	2
.	.	.	.	.	.
.	.	.	.	.	.
0	0	1	2	2	2
0	0	2	0	0	0
0	0	2	0	0	1
0	0	2	0	0	2
.	.	.	.	.	.
.	.	.	.	.	.
0	0	2	2	2	2
.	.	.	.	.	.
.	.	.	.	.	.
2	2	2	0	0	0
2	2	2	0	0	1
2	2	2	0	0	2
.	.	.	.	.	.
.	.	.	.	.	.
2	2	2	2	2	2

TABLE VIII

ALL POSSIBLE SET-RESET PAIRS
EXPRESSED IN DECIMAL NUMBERS

SET	RESET
0	0
0	1
0	2
.	.
.	.
0	26
1	0
1	1
1	2
.	.
.	.
1	26
2	0
2	1
2	2
.	.
.	.
2	26
.	.
.	.
26	0
26	1
26	2
.	.
.	.
26	26

TABLE IX

SIMPLIFIED SET-RESET PAIRS
EXPRESSED IN DECIMAL NUMBERS

SET	RESET
2	1
3	1
3	2
4	1
4	2
4	3
.	.
.	.
26	1
26	2
26	3
26	4
26	5
.	.
.	.
26	25

CHAPTER VIII

THE SIMPLEST SET OF SECONDARY VARIABLES

8.1 FLIP-FLOP ELEMENTS

Once a set-reset pair is chosen it can be used to set-reset a flip-flop element in order to obtain a secondary variable.

The flip-flop device is a memory element. Its output condition is dependent not only on the input but also on the previous output condition. Therefore, before using a flip-flop, we must know the initial condition of the output. A truth table of the flip-flop element is shown in Table X. If we let T(1,K), T(2,K) and Y(K-1), Y(K) express respectively the logic conditions of the set-reset signals at state K and the logic conditions of the flip-flop output at state K-1, and K, then Table X can be expressed as follows:

$$Y(K)=(Y(K-1))*(1-T(1,K))+(1-Y(K-1))*(1-T(1,K))*(T(2,K))$$

With the aid of this equation, we can develop a program to find all possible secondary variables involved in an operating system. Among all these possible secondary variables there are some redundant ones. The redundancy must be eliminated in order to reduce circuit complexity. A program which can be used to find the simplest set of secondary variables is developed in the next section.

8.2 THE SIMPLEST SET OF SECONDARY VARIABLES

In order to obtain the simplest set of secondary variables, we first

sort out those secondary variables which can differentiate the like ambiguous pairs and put them in a group. Those variables in the same group are considered to be equivalent variables. The sorting process is carried out as follows.

Let us assign each secondary variable a reference number. If there are n ambiguous pairs in a system, we consider the reference number of a secondary variable as an n-digit binary number in such a manner that the Ith digit is 1 provided that the two states of the Ith ambiguous pair can be differentiated between each other by this variable. Otherwise, the Ith digit is 0. In so doing, secondary variables which are equivalent possess the same reference number. Conversely, they have different reference numbers if they are not equivalent.

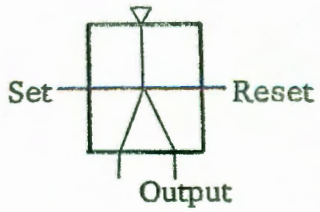
Next, we choose a secondary variable which contains the least component signals from each group of equivalent variables. All other variables excluded must be eliminated since they are redundant according to Section 2.6.

After the secondary variable among a group of equivalent variables is selected, construct the array MD(I,J) and execute the subprogram "SIMP". The simplest set of secondary variables can be achieved according to Chapter 4.

TABLE X

A TRUTH TABLE FOR A FLIP-FLOP ELEMENT

Set	Reset	Previous Output	Present Output
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1



CHAPTER IX

NON-AMBIGUOUS SYSTEMS

By executing the subprogram "SIMP", we can obtain the simplest combination of secondary variables to differentiate between the two states of the first n ambiguous pairs in each ambiguous group (Chapter 2, Theorem 1). If some ambiguities still exist, let us combine the secondary variables and the feedback signals as inputs and run the programs in Chapters 7 and 8. Again, we will find some additional secondary variables which are able to differentiate some of the remaining ambiguous states. By executing these three programs sequentially, we can obtain a sufficient number of secondary variables such that all previously ambiguous states will become non-ambiguous.

Now that sufficient secondary variables are found, we shall consider the feedback signals together with the secondary variables as the circuit inputs. A non-ambiguous relationship between the inputs and outputs can then be established.

CHAPTER X

OUTPUT EXPRESSIONS

10.1 INTRODUCTION

An output signal at a certain state can be obtained from a combination of input signals, i.e., an input combination at this state. If this input combination does not repeat itself at any other state where this output signal is "off", then this combination can be used to represent this output at this state. These types of input combinations are called possible output expressions.

A program is developed in this chapter which will find all possible output expressions and select a simplest set among these combinations in order to represent a control signal in an operating system.

10.2 ALL POSSIBLE OUTPUT EXPRESSIONS

From Section 7.2, we can see that there is a one-to-one correspondence between all possible combinations of input signals in a system in NI input signals and the integers 1,2,3,...,3**NI. These integers are assigned to their corresponding combinations as index numbers in the computer program.

In order to obtain all possible output expressions for a control signal at a certain state where this signal is "on" or "don't-care", we first look for all possible input combinations at this state. Next we shall

check the index number of each possible combination to see whether this combination repeats itself at any other states where the control signal is "off". After removing all combinations which appear at a certain "off" state, we can achieve all possible output expressions for this control signal at this state.

By repeating this process, we can obtain all possible output expressions for a control signal at every state where this control signal is "on" or "don't-care".

10.3 THE SIMPLEST SET OF OUTPUT EXPRESSIONS

Output expressions which can express a control signal at the same state or states are considered to be equivalent and only the one containing the least component signals is to be selected.

After the simplest output expression in each group of equivalent expressions is chosen, we can form an array MD(I,J) in a manner such that the MD(I,J) is the index number of the Ith expression at the Jth state. Then we can obtain the simplest set of output expressions for a control signal by executing the subprogram "SIMP". The above process is the same as that used in obtaining the simplest set of secondary variables mentioned in Sections 8.2 and 4.4.

10.4 "ON" AND "DON'T-CARE" STATES

In graphical design the output expressions for "don't-care" were

indicated by using underlines. Here, in the computer aided design, we assign expressions which are required for "on" states and for "don't-care" states only with the alphameric strings "ON" and "DONT", respectively. Therefore, in addition to finding the simplest combination of expressions, we also must classify these expressions. This can be done by executing the subprogram "SIMP" again to choose from all output expressions the expressions that stand for "on" states. Of course, those expressions that have not been selected are for "don't-care" states only.

10.5 OUTPUT EQUATIONS

In the computer printout, we use "1", "-1", and "0" to indicate the logic conditions "on", "off", and "don't-care" for each component signal of an output expression. Although the final control equations are not printed out, they can be obtained by transferring each output expression together with its associated alphameric string. For example, if the com-

puter printout reads:

THE OUTPUT FUNCTION IS

C1	CONDITION	F1	F3	F5	F7
	ON	0	0	-1	-1
	DONT	0	-1	1	0
	DONT	-1	0	0	-1
C2	CONDITION	F1	F3	F5	F7
	ON	0	0	-1	1
	ON	0	1	0	1
	DONT	1	0	1	0
C3	CONDITION	F1	F3	F5	F7
	ON	0	0	1	-1
	DONT	-1	0	-1	0
	DONT	0	-1	0	-1
C4	CONDITION	F1	F3	F5	F7
	ON	1	0	0	1
	ON	0	0	1	1
	DONT	0	1	-1	1

the output equations become:

C1=F6F8+F4F5+F2F8

C2=F6F7+F3F7+F1F5

C3=F5F8+F2F6+F4F8

C4=F1F7+F5F7+F3F6

The control circuit can be built based on these equations together with the set-reset signals needed for the secondary variables.

CHAPTER XI

SUMMARY

The essential element in the design of a sequential control circuit is the ability to differentiate between the ambiguous states contained in a circuit. In order to accomplish this, several synthesis techniques have been discussed. This thesis develops a method called the state diagram method. In this method, every two ambiguous states are made differentiable by some additional signals called secondary variables. Each secondary variable is obtained by the set-reset of a flip-flop element with a pair of input combinations. After a sufficient number of secondary variables are found such that there exist no more ambiguous pairs, a simple set of final logic equations can be obtained with the consideration of the logic "don't-care" and complementary conditions. The complete process of design using this method has been programmed on a digital computer.

In comparison with other methods, the state diagram synthesis contains several special features:

1. There is a definite routine to follow when designing a control circuit.
2. A simple set of final logic equations can be obtained, according to which a control circuit can be designed.
3. Possible circuit hazards can be located in advance and removed.

4. This synthesis is applicable to any type of sequential circuit of the fundamental mode.

5. This design method can be applicable to hydraulic, pneumatic, and electronic digital circuit designs with proper consideration of physical characteristics of respective components.

BIBLIOGRAPHY

1) Wickes, William E., Logic Design with Integrated Circuits, John Wiley & Sons, Inc., New York, 1969.

2) Hill, Fredrick J., and Peterson, Gerald R., Introduction to Switching Theory and Logical Design, Wiley & Sons, Inc., Salt Lake City, Utah, 1968.

3) Foster, K., and Parker, G.A., Fluidics-Components and Circuits, Wiley-Interscience, London, 1970, pp. 374-384.

4) Cheng, R.M.H. and Foster, K., "Systematic Method of Designing Fluidic-Pneumatic Control Circuits", Proceedings of the Institute of Mechanical Engineers, Vol. 186, 1972, pp. 401-408.

5) Cheng, R.M.H., and Foster, K., "A Computer-Aided Design Method Specially Applicable to Fluidic-Pneumatic Sequential Control Circuits", ASME Paper 70-WA/FLCS-17, 1970.

6) Standen, G.W., "Industrial Sequencing with Fluidics", Fluidics Quarterly, Vol. 2, No. 3, 1970, pp. 27-38.

7) Cheng, R.M.H., "Application of Fluidic Shift-Register Modules for Sequential Control of Pneumatic Sequential Circuits", Fifth Cranfield Fluidics Conference, Vol. 2, 1972, pp. F2-13 to F2-F28.

8) Cole, J.H., and Fitch, E.C., "Synthesis of Fluid Logic Circuits with Combined Feedback Input Signals", Fluidic Quarterly, Vol. 2, No. 5, 1970, pp. 14-21.

9) Chen, P.I. and Lee, Y.H., "State Diagram Synthesis of Fluidic Feedback Circuits", ASME Paper 73-WA/FLCS-2, 1973.

10) Stuart, Fredric, Fortran Programming, John Wiley & Sons, Inc., New York, 1969.

11) Rosenthal, Myron R., Numerical Methods in Computer Programming, Richard D. Irwin, Inc., Homewood, Illinois, 1966.

12) Hartkemeier, Harry P., Fortran Programming of Electronic Computers, Charles E. Merrill Books, Inc., Columbus, Ohio, 1966.

13) Organick, Elliott I., A Fortran IV Primer, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1966.

14) Humphery, Watts S., Switching Circuits with Computer Applications, McGraw-Hill Book Company, Inc. New York, 1958.

15) Richards, R.K., Digital Design, John Wiley & Sons, Inc., New York, 1971.

16) Phister, Montgomery, Logical Design of Digital Computers, John Wiley & Sons, Inc., Los Angeles, California, 1958.

17) Tedeschi, Frank P., and Scigliano, John A., Digital Computers & Logic Circuits, Glencoe Press, Beverly Hills, California, 1971.

18) Humphery, Eugene F., and Tarumoto, David H., Fluidics, Fluid Amplifier Associates, Incorporated, Ann Arbor, Michigan, 1968.

19) Conway, Arthur, A Guide to Fluidics, Macdonald & Co., (publishers) Ltd., New York, 1971.

20) Brown, Forbes T., Advances in Fluidics, The American Society of Mechanical Engineers, New York, 1967.

21) Belsterling, Charles A., Fluidic Systems Design, John Wiley & Sons, Inc., New York, 1971.

22) Fluidic Systems Design Guide, Imperial-Eastman Corporation, Chicago, Illinois, 1966.

23) Chen, P.I., and Lee, Y.H., "State Diagram Synthesis for Fundamental Mode Sequential Feedback Control Circuits", Fluidic State-Of-The-Art Symposium, Vol. 3, pp. 171-187.

24) Lewin, D.W., "A New Approach to the Design of Asynchronous Logic", The Radio and Electronic Engineer, Vol. 36, 1968, pp. 327-334.

25) Woods, R.L. and Reid, K.N., "A Procedure for the Logical Synthesis of Hybrid Control Systems", Journal of Dynamic Systems, Measurement, and Control, Vol. 93, 1971, pp. 227-232.

26) Caywood, James A., "Digital Controls Help Develop Hydraulics for Backhoe", Hydraulics & Pneumatics, No. 22, 1969, pp. 108-111.

27) Huff, P.C., "Sequence Control Circuits Simplify Air Logic Design", Hydraulics & Pneumatics, No. 25, 1972, pp. 64-67.

28) Bronsard, Joseph A., "Fluidic Circuit Controls Board Unloader and Stacker", Hydraulics & Pneumatics, No. 25, 1972, pp. 118-121.

29) Maroney, G.E. and Fitch, E.C., "A Synthesis Technique for Fluid Logic Control Networks", Fluidic Quarterly, Vol 3., No. 1, 1971, pp. 38-46.

30) Fitch, E.C. and Maroney, G.E., "Analysis of Fluid Logic Circuit Networks", Fluidic Quarterly, Vol. 3., No. 1, 1971, pp. 47-54.

31) Zissos, D., "Design Algorithms for Fluidic Circuits", Fluidics Quarterly, Vol. 3, No. 4, 1971, pp. 45-55.

32) Maroney, G.E. and Fitch, E.C., "The Mathematical Synthesis and Analysis of Fluid Logic Networks", Fluidic Quarterly, Vol. 4, No. 3, 1972, pp. 44-57.

APPENDIX A

COMPUTER PROGRAM AND PRINTOUT

```
*P.S.U.   COMPUTER CENTER*

// $            YAUHWANG LEE

// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
C     ************************************************************************
C
C     SUBPROGRAM - 'SIMP'
C
      SUBROUTINE SIMP
      INTEGER A(10),B(10),E(120),NA(10),MA(10),MD(10,31)
      COMMON A,E,NA,NE,NN,MD
      NE=NN+1
      DO 180 I=1,NN
180   B(I)=1
190   N=0
      M=0
      DO 230 I=1,NN
      LA=B(I)
      IF(I-NN) 200,220,200
200   J=I+1
      L=1
      DO 210 K=J,NN
      LB=B(K)
210   L=L*(MD(I,LA)-MD(K,LB))
      IF(L) 220,230,220
220   N=N+1
      K=MD(I,LA)
      M=M+E(K)
      MA(N)=MD(I,LA)
230   CONTINUE
      IF(NE-N) 270,240,250
240   IF(ME-M) 270,270,250
250   NE=N
      ME=M
      DO 260 I=1,N
260   NA(I)=MA(I)
270   I=NN
280   J=1/(1+A(I)-B(I))
      B(I)=1+B(I)*(1-J)
      I=I-1
      IF((I+1)*J-1) 190,290,280
290   RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR SIMP
 COMMON     452  VARIABLES      32  PROGRAM     272
```

```
RELATIVE ENTRY POINT ADDRESS IS 0022 (HEX)

END OF COMPILATION

// DUP

*STORE      WS  UA  SIMP
CART ID 0001   DB ADDR  368A   DB CNT   0011

// FOR
* IOCS (CARD, 1403 PRINTER)
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      INTEGER A(10),C(8,10),D(511),E(120),F(10,10),T(2,10),Y(15,11)
      DIMENSION VA(10),VB(120),WA(10),WB(10),WC(2,15,5),WD(10,31),V(7)
      COMMON A,E,VA,VB,NV,WD
C     ****************************************************************
C
C     PROGRAM I - OBTAIN LOGIC CONDITIONS OF FEEDBACKS AND CONTROLS
C
      READ(2,140) (V(I),I=1,7)
100   WRITE(5,150)
      READ(2,160) NC,NS
      IF(NC) 980,980,105
105   NF=2**NC
      WRITE(5,170) NC,NS,(V(I),I,I=1,NF)
      DO 110 I=1,NF,2
      READ(2,160) (F(I,J),J=1,NS)
      DO 110 J=1,NS
110   F(I+1,J)=1-F(I,J)
      DO 130 I=1,NS
      K=I+1-NS*(I/NS)
      A(I)=0
      DO 120 J=1,NF
      C(J,I)=(F(J,I)+F(J,K)+1)*(1-F(J,K))
120   A(I)=A(I)+J*C(J,I)/2
130   WRITE(5,180) V(2),A(I),I,(F(J,I),J=1,NF)
140   FORMAT(7A4)
150   FORMAT(1H1,///, 72('*'))
160   FORMAT(3012)
170   FORMAT(/,' THIS IS A',I2,'-CYLINDER',I3,'-STATE SYSTEM WHICH OPERA
     1TES AS FOLLOW ..',/,T53,'CONTROL',T6,'STATE',3X,10(2X,A1,I1))
180   FORMAT(T55,A1,I1,T7,I2,5X,10I4)
C     ****************************************************************
C
C     PROGRAM II - FIND AMBIGUOUS STATE PAIRS
C
      ND=0
      DO 230 I=2,NS
      DO 230 J=I,NS
      K=0
      L=0
190   M=I-1-K+NS*((NS+K-1-I)/NS)
      N=J-K+NS*((NS+K-J)/NS)
      DO 200 NN=1,NF,2
200   L=L+IABS(F(NN,M)-F(NN,N))
      IF((1/(L+1))*(K+IABS(A(M)-A(N)))*(2*ND-11) 210,230,220
210   WRITE(5,270)
220   WRITE(5,280) M,N
```

```
      K=K+1
      ND=ND+1
      MA(ND)=M
      MB(ND)=N
      GOTO 190
230   CONTINUE
      I=6+1/(1+ND)
      WRITE(5,250) V(I)
      NG=0
      IF(ND) 650,650,240
240   WRITE(5,260) (I,I=1,NS)
250   FORMAT(/,1X,A4,' SECONDARY VARIABLE(S) ARE REQUIRED')
260   FORMAT(/,7X,'SET',7X,'RESET',5X,12I4)
270   FORMAT(/,' SOME AMBIGUOUS PAIR(S) EXIST IN THIS SYSTEM ..')
280   FORMAT(7X,I4,4X,'AND',I5)
C     ***********************************************************************
C
C     PROGRAM III - OBTAIN ALL POSSIBLE SET-RESET PAIRS
C
      NI=NC
290   NP=1
      LA=NC+NG-NI
      NI=NC+NG
      NE=2*NI
      I=2**ND-1
      JJ=3**NI-1
      DO 300 J=1,ND
300   A(J)=0
      DO 310 J=1,I
310   D(J)=0
      DO 530 I=2,JJ
      DO 530 J=2,I
      DO 320 K=1,2
      DO 320 L=1,NS
320   T(K,L)=1
      II=NE
      LB=0
      DO 350 K=1,NI
      DO 350 L=1,2
      MC(L,NP,K)=((J-I-1)*(L-1)+I)/3**(K-1)-3*(((J-I-1)*(L-1)+I)/3**K)
      IF(MC(L,NP,K)) 350,350,330
330   M=MC(L,NP,K)+2*(K-1)
      LB=LB+K/(NI-LA+1)
      DO 340 N=1,NS
340   T(L,N)=T(L,N)*F(M,N)
350   II=II-1/(1+MC(L,NP,K))
      IF(LA*(1/(1+LB))) 360,360,530
360   K=0
      DO 370 L=1,NS
370   K=K+T(1,L)*T(2,L)
      IF(K) 530,380,530
C     ***********************************************************************
C
C     PROGRAM IV - FIND ALL POSSIBLE SECONDARY VARIABLES
C
380   Y(NP,1)=-1
390   Y(NP,1)=Y(NP,1)+1
      DO 400 L=1,NS
      M=L+1-NS*(L/NS)
```

```
400   Y(NP,L+1)=Y(NP,L)*(1-T(2,M))+(1-Y(NP,L))*(1-T(2,M))*T(1,M)
      IF(Y(NP,1)-Y(NP,NS+1)) 390,410,390
410   DO 420 L=1,ND
      M=MA(L)
      N=MB(L)
      NA(L)=1-IABS(Y(NP,M)+Y(NP,N)-1)
420   K=K+(2**(L-1))*NA(L)
      IF(K) 530,530,430
430   IF(D(K)) 530,470,440
440   NN=D(K)
      IF(E(NN)-II) 530,530,450
450   DO 460 L=1,NS
460   Y(NN,L)=Y(NP,L)
      GOTO 510
470   NN=NP
      D(K)=NN
      DO 490 L=1,ND
      IF(NA(L)) 490,490,480
480   A(L)=A(L)+1
      M=A(L)
      MD(L,M)=NN
490   CONTINUE
510   E(NN)=II
      DO 520 K=1,2
      DO 520 L=1,NI
520   MC(K,NN,L)=MC(K,NP,L)+(1-1/(1+MC(K,NP,L)))*2*(L-1-NC*(1/(1+NC/L)))
      NP=NP+NN/NP
530   CONTINUE
C     ***************************************************************
C
C     PROGRAM V - FIND THE SIMPLEST SET OF SECONDARY VARIABLES
C
      NN=0
      I=ND
      ND=0
      DO 570 J=1,I
      IF(A(J)) 540,560,540
540   NN=NN+1
      A(NN)=A(J)
      K=A(J)
      DO 550 L=1,K
550   MD(NN,L)=MD(J,L)
      GOTO 570
560   ND=ND+1
      MA(ND)=MA(J)
      MB(ND)=MB(J)
570   CONTINUE
      CALL SIMP
      DO 610 I=1,NE
      J=NA(I)
      NG=NG+1
      K=2*NG-1
      L=NF+K
      DO 580 M=1,NS
      F(L,M)=Y(J,M)
580   F(L+1,M)=1-Y(J,M)
      IF(NI-NC) 590,590,600
590   WRITE(5,620) V(3),K,(V(1),MC(1,J,M),M=1,NC)
      WRITE(5,630) (V(1),MC(2,J,M),M=1,NC)
```

```
      GOTO 610
600   M=NC+1
      WRITE(5,620)V(3),K,(V(1),MC(1,J,N),N=1,NC),(V(3),MC(1,J,N),N=M,NI)
      WRITE(5,630) (V(1),MC(2,J,N),N=1,NC),(V(3),MC(2,J,N),N=M,NI)
610   WRITE(5,640) (Y(J,N),N=1,NS)
      IF(ND) 290,650,290
620   FORMAT(2X,A1,I1,T8,5(A1,I1))
630   FORMAT(1H+,T18,5(A1,I1))
640   FORMAT(1H+,T28,12I4)
C     ********************************************************************
C
C     PROGRAM VI - FIND ALL POSSIBLE OUTPUT EXPRESSIONS
C
650   WRITE(5,840)
      NK=2*NG
      NI=NC+NG
      II=2**NI-1
      JJ=3**NI-1
      DO 940 I=1,NF
      NP=0
      NG=0
      NN=0
      DO 660 L=1,JJ
660   D(L)=0
      DO 710 J=1,3
      DO 710 K=1,NS
      IF((C(I,K)+J-1)*(C(I,K)+J-4)) 710,670,710
670   NN=NN+1-1/J
      NG=NG+(J-1)*(3-J)
      MM=0
      DO 710 L=1,II
      ND=0
      NE=0
      DO 680 M=1,NI
      ND=ND+L/2**(M-1)-2*(L/2**K)
680   NE=NE+(L/2**(M-1)-2*(L/2**M))*(F(2*K-1,K)+1)*(3**(M-1))
      N=D(NE)
      D(NE)=1-2*(1/J+(IABS(N)-N))
      IF((1/(1+IABS(D(NE))-D(NE)))*(1-2*N)) 700,710,690
690   NP=NP+1
      NB(NP)=NE
      D(NE)=NP
      E(NP)=ND
      GOTO 705
700   D(NE)=N
705   MM=MM+1
      A(NN)=MM
      MD(NN,MM)=D(NE)
710   CONTINUE
C     ********************************************************************
C
C     PROGRAM VII - OBTAIN THE SIMPLEST SET OF OUTPUT EXPRESSIONS
C
      DO 740 J=1,NP
      N=0
      DO 730 K=1,NN
      MM=A(K)
      M=1
      DO 720 L=1,MM
```

```
720     M=M*(1-1/(IABS(J-MD(K,L))+1))
730     N=N+(1-M)*(2**(K-1))
740     D(J)=N
        ND=NP-1
        DO 800 J=1,ND
        IF(D(J)) 800,800,750
750     MM=J
        NE=J+1
        DO 795 K=NE,NP
        IF(D(MM)-D(K)) 795,760,795
760     IF(E(MM)-E(K)) 770,770,780
770     N=K
        GOTO 790
780     N=MM
        MM=K
790     D(N)=0
795     CONTINUE
800     CONTINUE
        DO 830 J=1,NN
        M=0
        MM=A(J)
        DO 820 K=1,MM
        L=MD(J,K)
        IF(D(L)) 820,820,810
810     M=M+1
        MD(J,M)=L
820     CONTINUE
830     A(J)=M
        CALL SIMP
840     FORMAT(/,' THE OUTPUT FUNCTION IS ..')
C       ************************************************************************
C
C       PROGRAM VIII - PRINT OUT THE OUTPUT FUNCTION
C
        IF(NK) 845,845,850
845     WRITE(5,950) V(2),I,(V(1),J,J=1,NF,2)
        GOTO 860
850     WRITE(5,950) V(2),I,(V(1),J,J=1,NF,2),(V(3),J,J=1,NK,2)
860     ND=NE
        DO 870 K=1,NE
870     D(K)=NA(K)
        M=0
        DO 910 N=1,NG
        NN=1
        NP=A(N)
        DO 900 L=1,NP
        DO 880 K=1,NE
880     NN=NN*(1-1/(IABS(NA(K)-MD(N,L))+1))
        IF(NN) 900,890,900
890     M=M+1
        MD(N,M)=MD(N,L)
900     CONTINUE
910     A(N)=M
        NN=NG
        CALL SIMP
        DO 940 J=1,ND
        L=1
        DO 920 K=1,NE
920     L=L*(1-(1/(IABS(D(J)-NA(K))+1)))
```

```
      L=L+4
      M=D(J)
      M=NB(M)
      DO 930 K=1,NI
      N=M/3**(K-1)-3*(M/3**K)
930   A(K)=3*(N/2)-N
940   WRITE(5,960) V(L),(A(M),M=1,NI)
      WRITE(5,970)
950   FORMAT(2X,A1,I1,3X,'CONDITION',2X,30(1X,A1,I1))
960   FORMAT(9X,A4,5X,20I3)
970   FORMAT(//, 72('*'))
      GOTO 100
980   CALL EXIT
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS
 IOCS

CORE REQUIREMENTS FOR
 COMMON    452  VARIABLES   1216  PROGRAM   3402

END OF COMPILATION

// XEQ
```

****

THIS IS A 4-CYLINDER 10-STATE SYSTEM WHICH OPERATES AS FOLLOW ..

STATE	F1	F2	F3	F4	F5	F6	F7	F8	CONTROL
1	1	0	1	0	1	0	1	0	C1
2	0	1	1	0	1	0	1	0	C5
3	0	1	1	0	0	1	1	0	C7
4	0	1	1	0	0	1	0	1	C2
5	1	0	1	0	0	1	0	1	C8
6	1	0	1	0	0	1	1	0	C3
7	1	0	0	1	0	1	1	0	C7
8	1	0	0	1	0	1	0	1	C6
9	1	0	0	1	1	0	0	1	C4
10	1	0	1	0	1	0	0	1	C8

NO SECONDARY VARIABLE(S) ARE REQUIRED

THE OUTPUT FUNCTION IS ..

		F1	F3	F5	F7
C1	CONDITION				
	ON	0	0	1	1
	DONT	-1	0	0	1
C2	CONDITION	F1	F3	F5	F7
	DONT	1	0	-1	0
	ON	0	0	0	-1
C3	CONDITION	F1	F3	F5	F7
	ON	1	0	-1	1
	DONT	0	-1	-1	0
C4	CONDITION	F1	F3	F5	F7
	DONT	-1	0	0	0
	DONT	0	1	0	-1
	ON	0	0	1	0
C5	CONDITION	F1	F3	F5	F7
	ON	-1	0	0	0
	DONT	0	1	-1	0
	DONT	0	-1	0	1
C6	CONDITION	F1	F3	F5	F7
	ON	0	-1	0	-1
	DONT	1	0	1	0
C7	CONDITION	F1	F3	F5	F7
	ON	-1	0	-1	0
	ON	0	-1	0	0
C8	CONDITION	F1	F3	F5	F7
	DONT	-1	0	1	0
	ON	1	1	0	0

****

```
**************************************************************************

THIS IS A 2-CYLINDER  8-STATE SYSTEM WHICH OPERATES AS FOLLOW ..
    STATE      F1   F2   F3   F4                          CONTROL
      1         1    0    1    0                            C1
      2         0    1    1    0                            C3
      3         0    1    0    1                            C4
      4         0    1    1    0                            C2
      5         1    0    1    0                            C3
      6         1    0    0    1                            C1
      7         0    1    0    1                            C2
      8         1    0    0    1                            C4

SOME AMBIGUOUS PAIR(S) EXIST IN THIS SYSTEM ..
          1     AND     5
          2     AND     4
          3     AND     7
          6     AND     8

SOME SECONDARY VARIABLE(S) ARE REQUIRED

      SET         RESET          1   2   3   4   5   6   7   8
 Y1   F1F4        F2F3           1   0   0   0   0   1   1   1
 Y3   F2F4        F1F3           0   0   1   1   0   0   1   1

THE OUTPUT FUNCTION IS ..
 C1   CONDITION    F1 F3 Y1 Y3
        ON          0  0  1 -1
        DONT       -1  0  0 -1
        DONT        0 -1 -1  0
 C2   CONDITION    F1 F3 Y1 Y3
        ON          0  1  0  1
        DONT        1  0 -1  0
        DONT        0  0  1  1
 C3   CONDITION    F1 F3 Y1 Y3
        ON          0  0 -1 -1
        DONT        0 -1  0 -1
        DONT       -1  0  1  0
 C4   CONDITION    F1 F3 Y1 Y3
        ON          1  0  0  1
        DONT        0  1  1  0
        ON          0  0 -1  1

**************************************************************************
```

```
**********************************************************************

THIS IS A 3-CYLINDER  8-STATE SYSTEM WHICH OPERATES AS FOLLOW ..
    STATE      F1  F2  F3  F4  F5  F6                  CONTROL
      1         1   0   1   0   1   0                    C1
      2         0   1   1   0   1   0                    C2
      3         1   0   1   0   1   0                    C1
      4         0   1   1   0   1   0                    C3
      5         0   1   0   1   1   0                    C2
      6         1   0   0   1   1   0                    C5
      7         1   0   0   1   0   1                    C6
      8         1   0   0   1   1   0                    C4

SOME AMBIGUOUS PAIR(S) EXIST IN THIS SYSTEM ..
          2    AND    4
          1    AND    3
          6    AND    8

SOME SECONDARY VARIABLE(S) ARE REQUIRED

      SET         RESET          1   2   3   4   5   6   7   8
 Y1   F0F0F6      F2F0F0         1   0   0   0   0   0   1   1
 Y3   F1F0F0Y2    F0F0F6Y0       0   0   1   1   1   1   0   0

THE OUTPUT FUNCTION IS ..
 C1    CONDITION    F1 F3 F5 Y1 Y3
         ON          1  1  0  0  0
         DONT        0  1  0  0  1
 C2    CONDITION    F1 F3 F5 Y1 Y3
         ON         -1  0  0  0 -1
         ON          0 -1  0  0  0
 C3    CONDITION    F1 F3 F5 Y1 Y3
         ON         -1  0  0  0  1
         DONT        0 -1  0 -1  0
         DONT        0  0 -1  0  0
 C4    CONDITION    F1 F3 F5 Y1 Y3
         ON          0  0  1  0 -1
         DONT        1  1  0  0  0
 C5    CONDITION    F1 F3 F5 Y1 Y3
         ON          1 -1  0 -1  0
 C6    CONDITION    F1 F3 F5 Y1 Y3
         DONT        0  1  0  0  0
         DONT       -1  0  0  0  0
         ON          0  0  0  1  0

**********************************************************************
```

APPENDIX B

FLOWCHARTS AND DATA CARDS

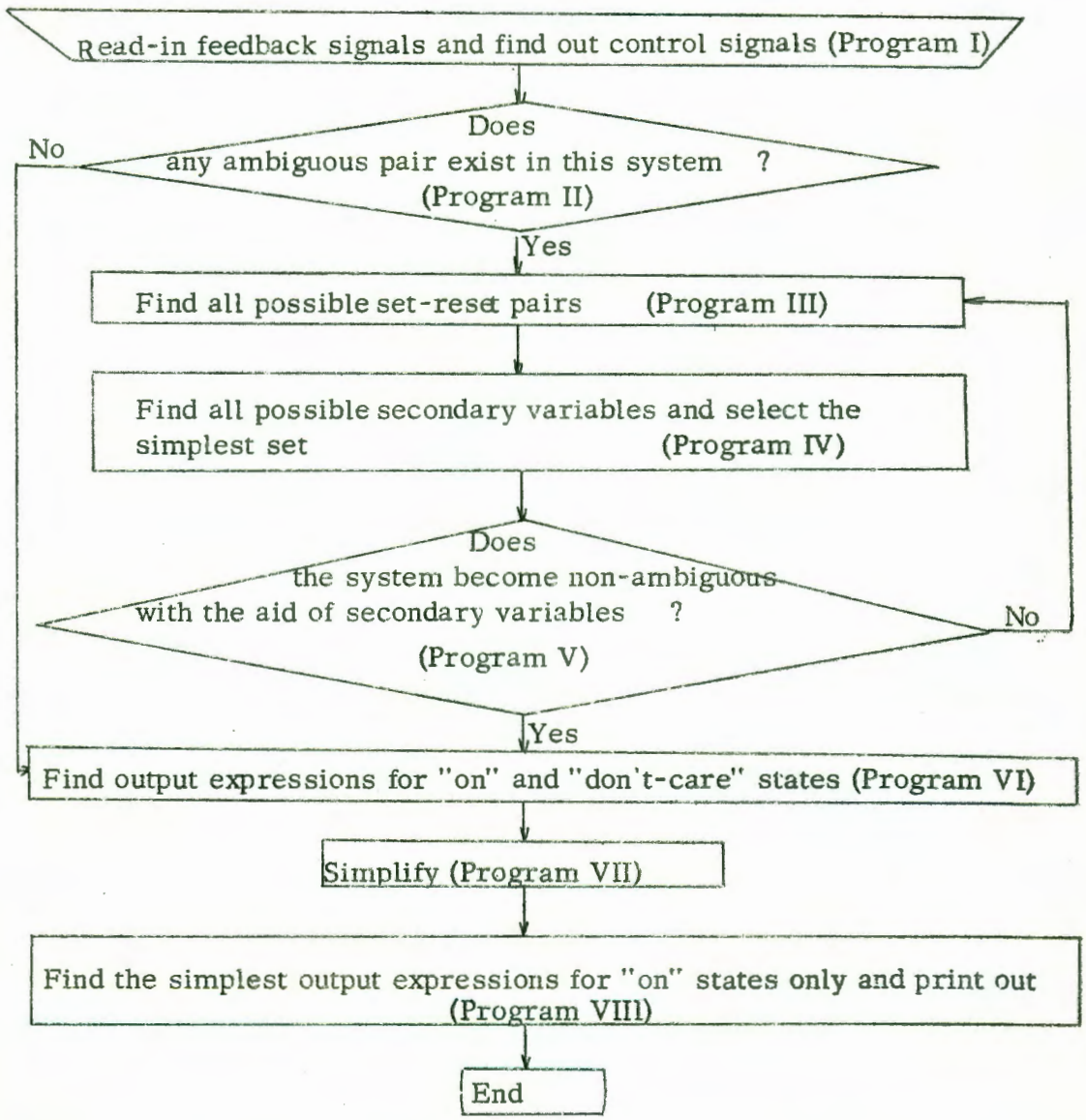


Figure 16. Simplified Flowchart for the Complete Computer Program

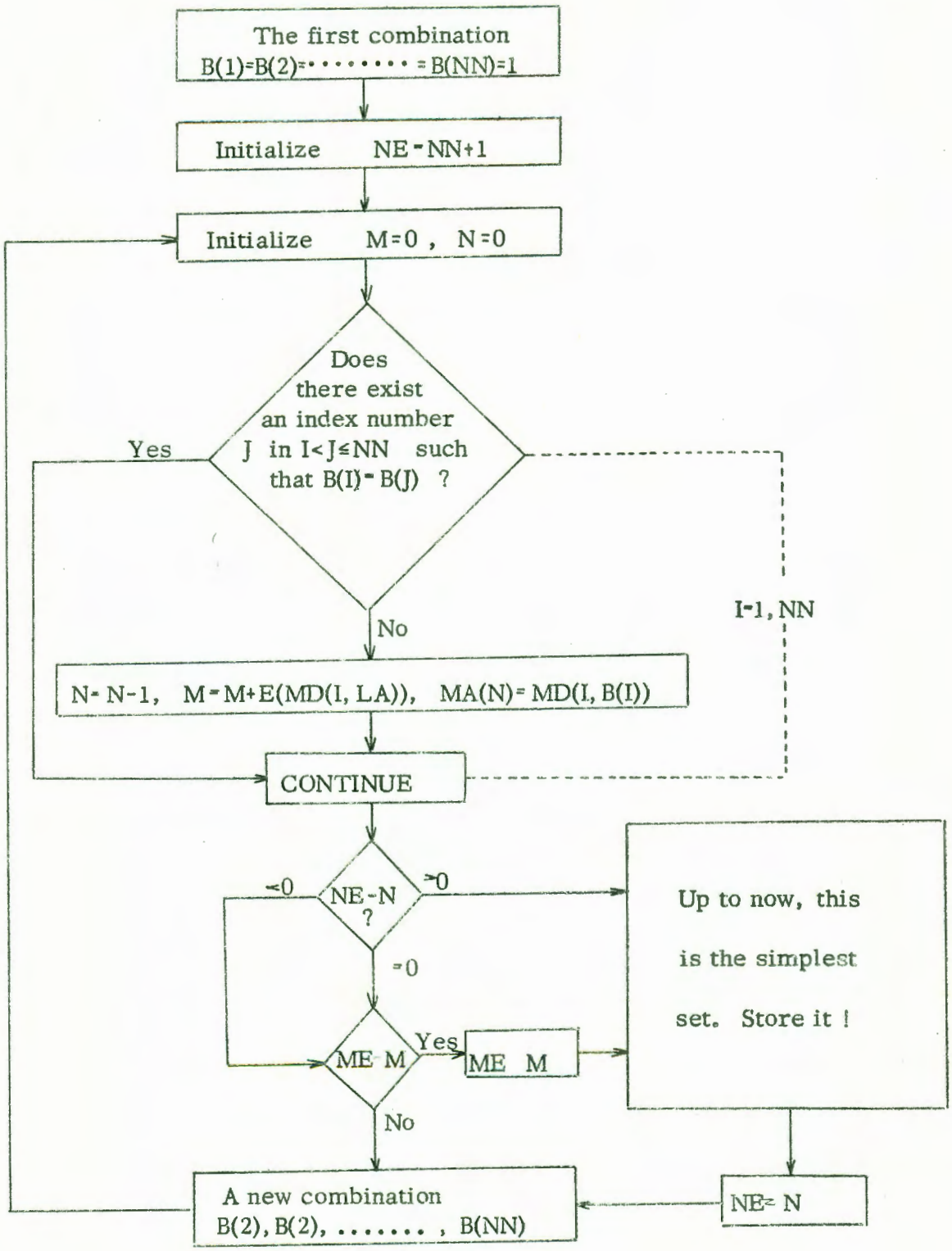


Figure 17. Flowchart for the subprogram "SIMP"

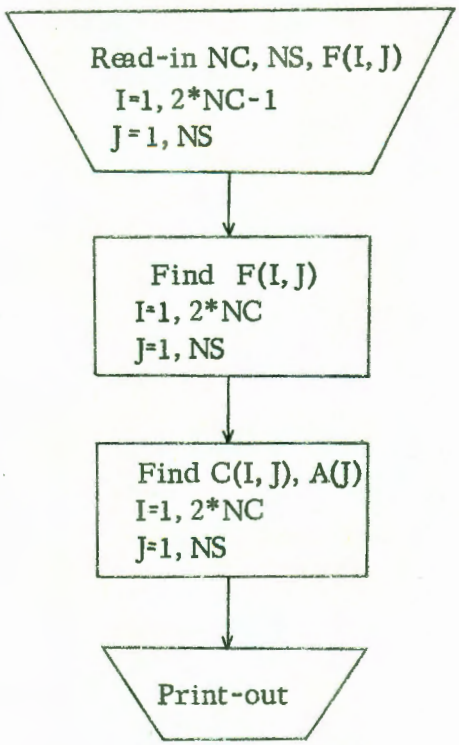


Figure 18. Flowchart of Program I

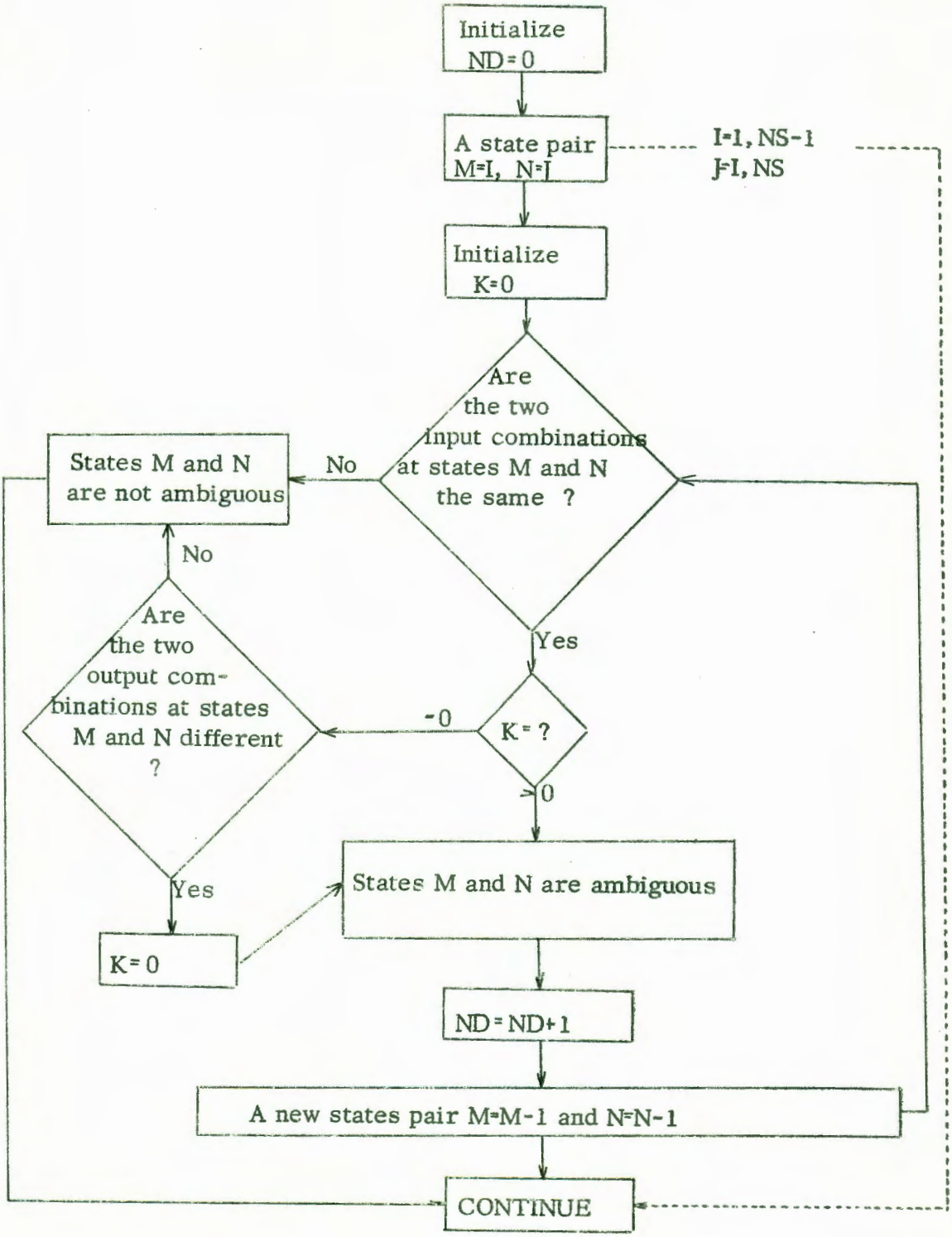


Figure 19. Flowchart of Program II

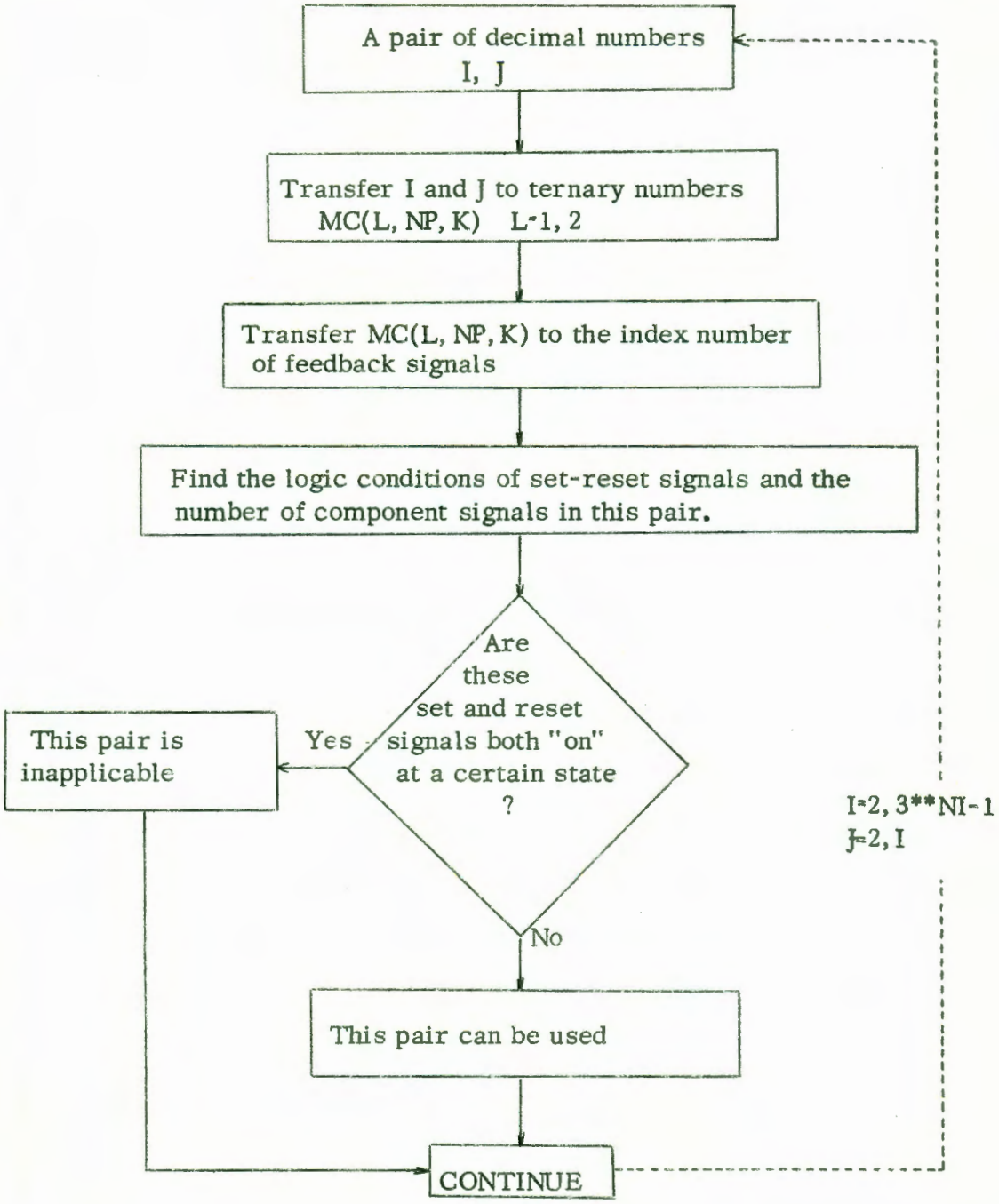


Figure 20. Flowchart of Program III

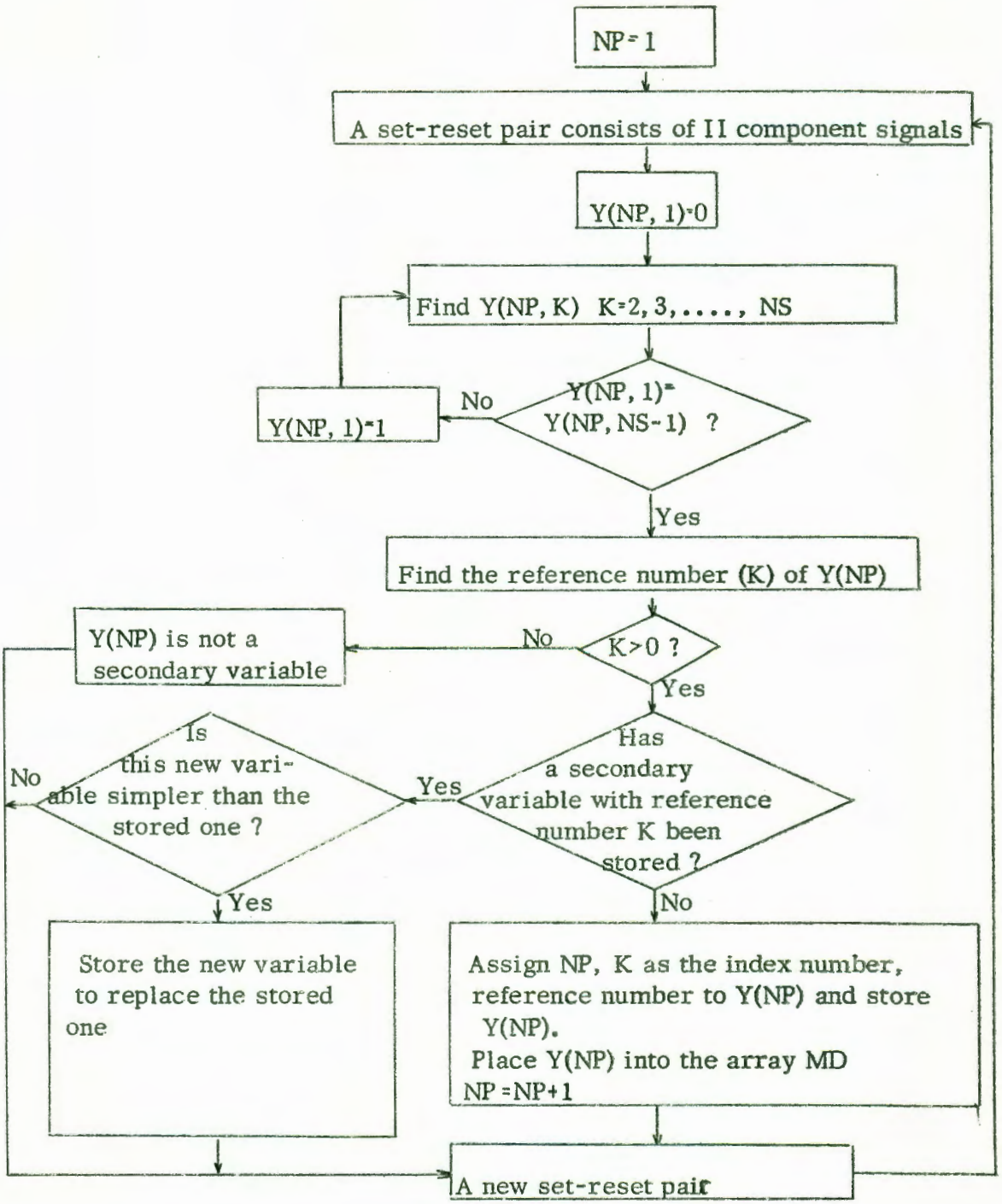


Figure 21. Flowchart of Program IV

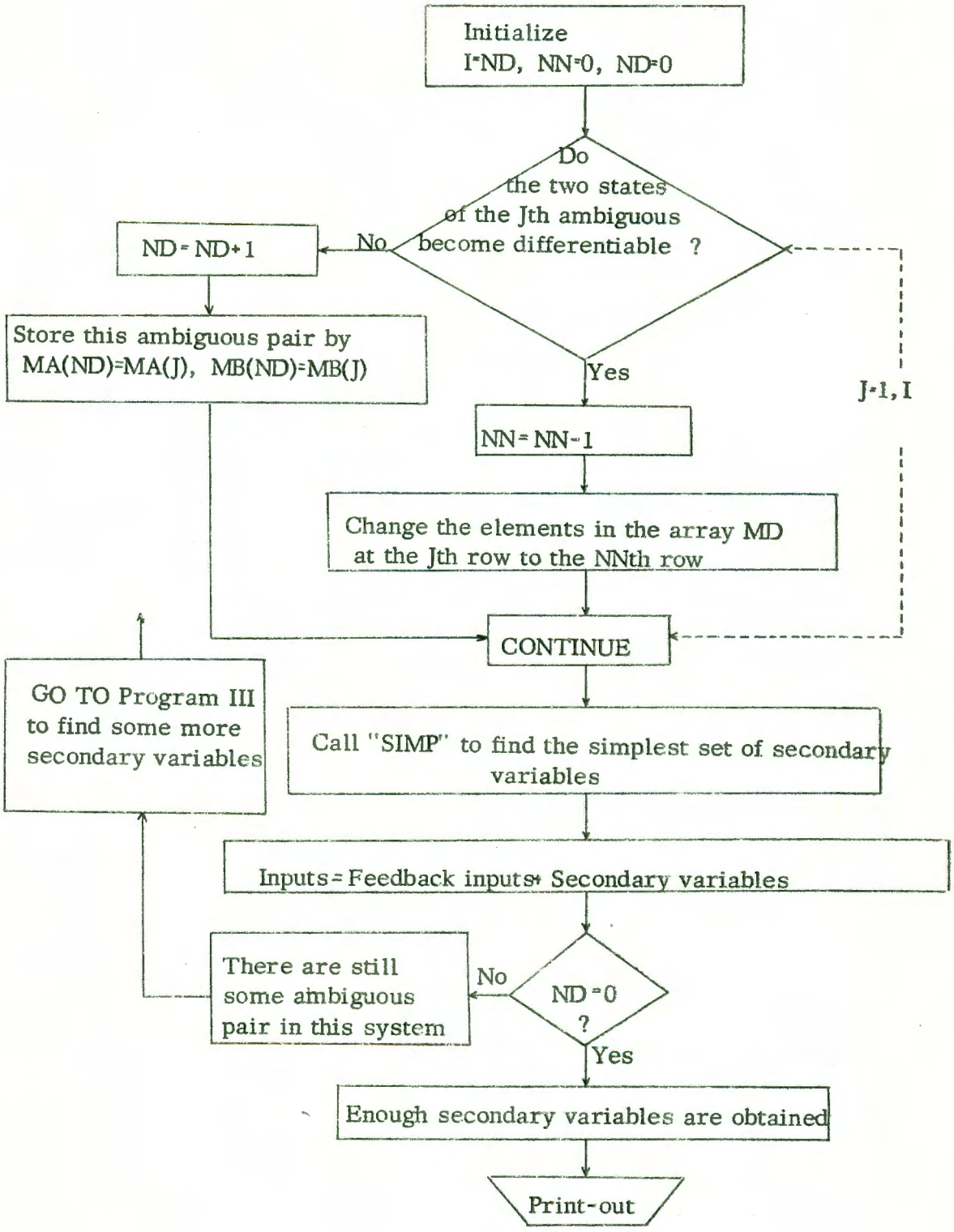


Figure 22. Flowchart of Program V

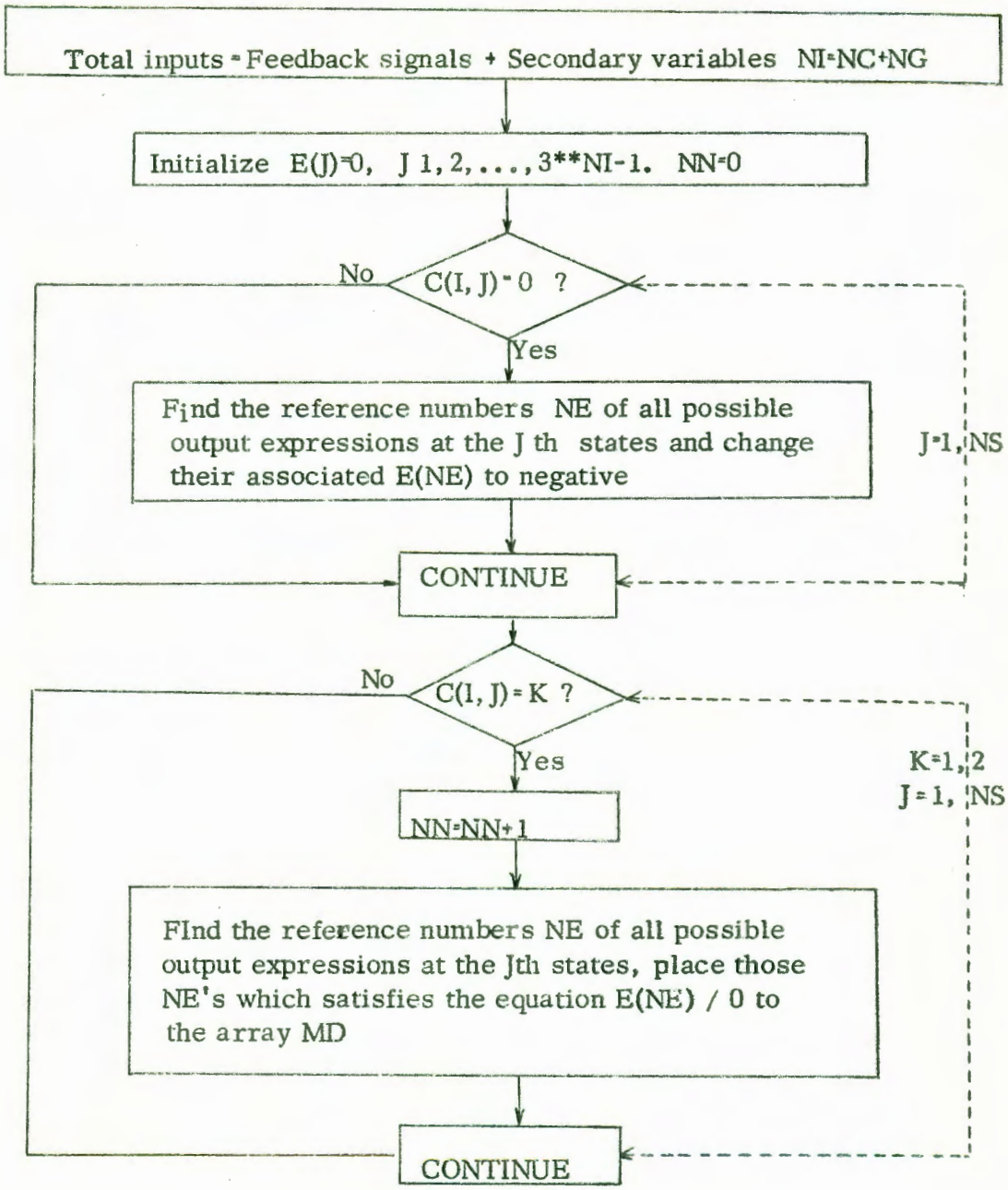


Figure 23. Flowchart of Program VI

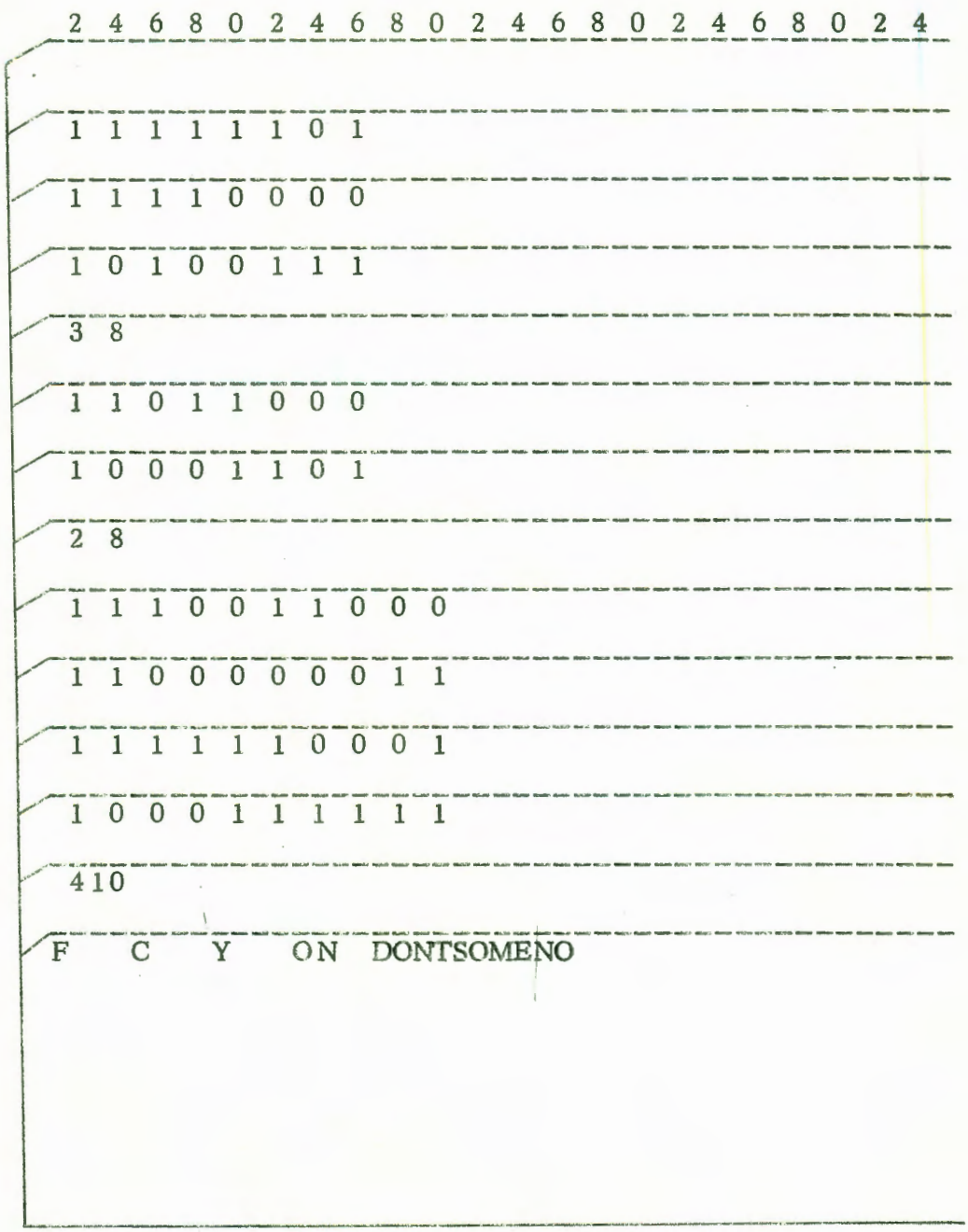


Figure 24. Data Cards

APPENDIX C

VARIABLES CONTAINED IN THE PROGRAM

A (I)	The index number of the "on" control signal at the Ith state (Programs I and II only).
A (I)	The number of elements in the Ith row of the array MD (I,J) (except Programs I and II).
C (I,J)	The logic condition of the Ith control signal at the Jth state.
D (I)	The number of input signals in the set-reset pair for the Ith secondary variable (except Programs VI,VII,VIII).
D (I)	A function representing whether an output expression with index number I can be used to express an output signal (Programs VI,VII, and VIII).
E (I) J	A function used to relate the reference number I of a secondary variable to the index number J (Programs III and IV).
E (I)	The number of input signals in the set-reset pair for the Ith secondary variable (Program V).
E (I)	The number of signals in an output expression with index number I (Programs VI,VII,VIII).

F(I,J)	The logic condition of the Ith input signal at the Jth state.
NA(I)	A function representing a possible secondary variable that can differentiate between the two states of the Ith ambiguous state pair (Program IV).
NA(I)	The index number of a secondary variable/input combination which is chosen for the simplified set of necessary secondary variables/input combinations (Programs VI,VII,VIII, and sub-program).
NB(I)=J	A function relating the index number I of an output expression to its reference number J.
NC	The number of cylinders/independent feedback signals.
ND	The number of ambiguous state pairs (except Programs VI,VII,VIII).
NE	The number of total input signals (=2*NI) (except Programs VI,VII,VIII).
NF	The number of total feedback signals (=2*NC).
NG	The number of independent secondary variables (except Programs VI,VII,VIII).

NI	The number of total independent input signals (=NC+NG).
NN	The row number of the array MD(I,J).
NP	The number of all possible secondary variables (except Programs VI,VII,VIII).
NS	The number of operating states.
MA(I),MB(I)	Two state numbers of the Ith ambiguous state pair. (Main program only.)
MC(1,I,J)-MC(2,I,J)	The index number of the Jth component signal in a set-reset pair for the Ith possible secondary variable.
MD(I,J)	The array.
T(1,I)-T(2,I)	The logic conditions of a set-reset pair at the Ith state.
V(1),V(2),...,V(7)	The alphameric strings "F", "C", "Y", "ON", "DONT", "SOME", and "ON".
Y(I,J)	The logic condition of the Ith secondary variable at the Jth state.

APPENDIX D

DIMENSIONS OF THE SUBSCRIPTED VARIABLES CONTAINED IN THE PROGRAM

Before we define the dimensions of the subscripted variables, let us assume the number of ambiguous pairs to be LL; the number of possible secondary variables to be MM; the number of "on" and "don't-care" states of an output signal to be II, and the number of possible output expressions to be JJ.

With these numbers we let AA=max(LL,NS), BB=max(AA,2*NI), CC=max(MM,JJ,2**LL-1), DD=max(2*NI,2*II-1), and EE=max(MM,JJ). The dimensions of these subscripted variables can then be defined as:

A	(AA)
B	(BB)
C	(NF,NS)
D	(DD)
E	(CC)
F	(2*NI,NS)
NA	(AA)
NE	(JJ)
MA,MB	(LL),(LL) in main program
	(AA),(AA) in subprogram
MC	(2,LL,2*NI)

MD	(AA ,EE)
T	(2 ,NS)
V	(5)
Y	(LL ,NS-1)

APPENDIX E

STATE DIAGRAMS, OUTPUT EQUATIONS, AND CONTROL CIRCUIT DIAGRAMS OF HYPOTHETICAL SYSTEMS 1 AND 2

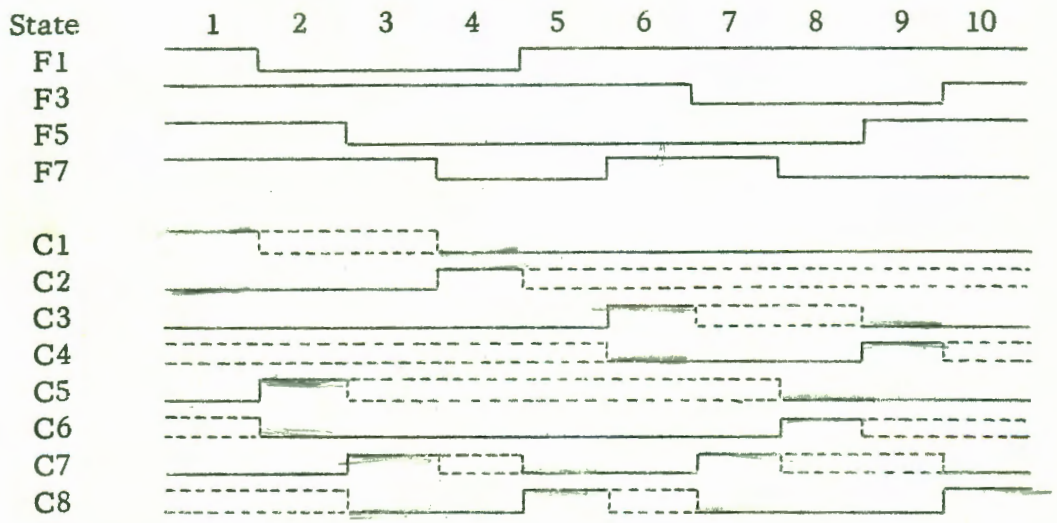


Figure 25. State diagram of System 1

Output equations of System 1

$C1 = F5F7X1\overline{X2} = \overline{\overline{F5} + \overline{F7} + \overline{X1} + X2}$

$C2 = \overline{F7} + X2$

$C3 = F1\overline{F5}F7\overline{X2} = \overline{\overline{F1} + F5 + \overline{F7} + X2}$

$C4 = F5 + X2$

$C5 = \overline{F1}\overline{X2}$

$C6 = \overline{F3}\overline{F7} + X2$

$C7 = (\overline{F1}\overline{F5} + \overline{F3})\overline{X2}$

$C8 = F1F3 + X2$

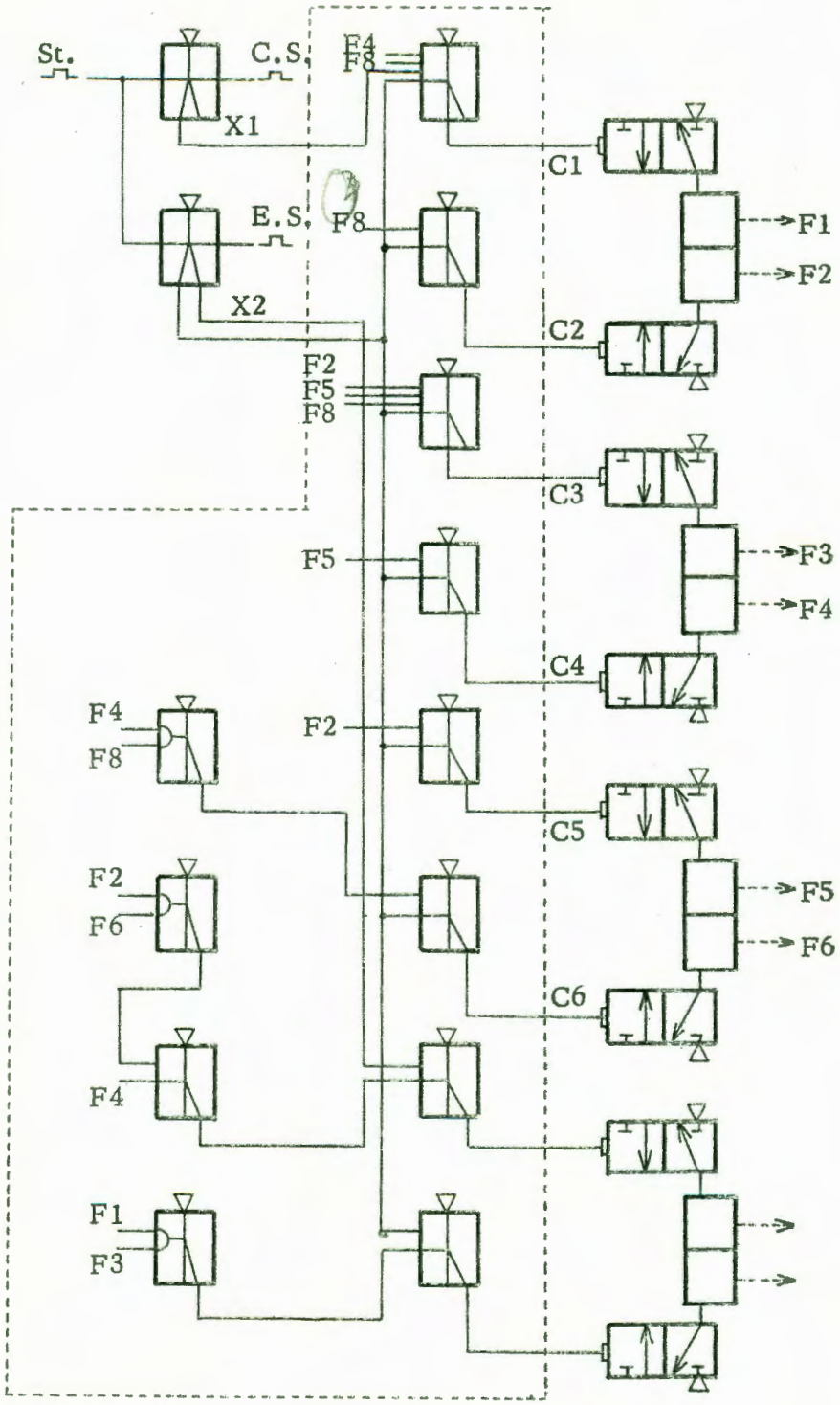


Figure 26. Control circuit diagram of System 1

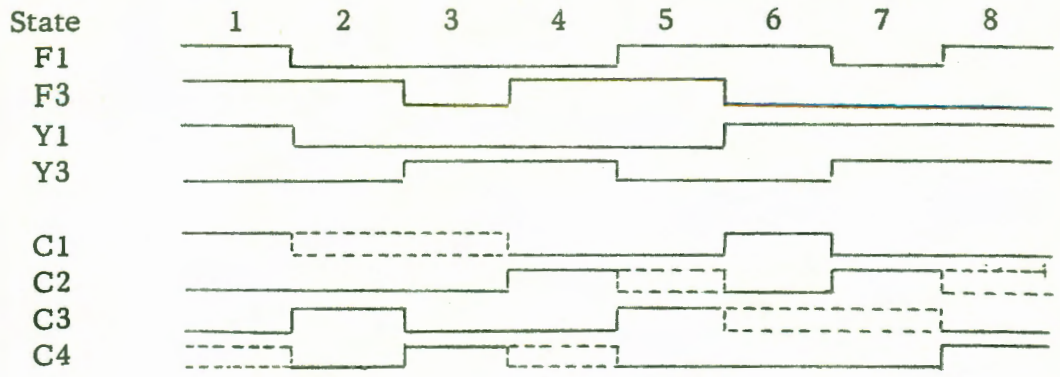


Figure 27. State diagram of System 2

Output equations of System 2

$C1=\overline{C2}(X1+\overline{F3})$
$C2=F3Y3+F1\overline{Y1}+Y1Y3+X2$
$C3=C4$
$C4=F1Y3+F3Y1+\overline{Y1}Y3+X2$

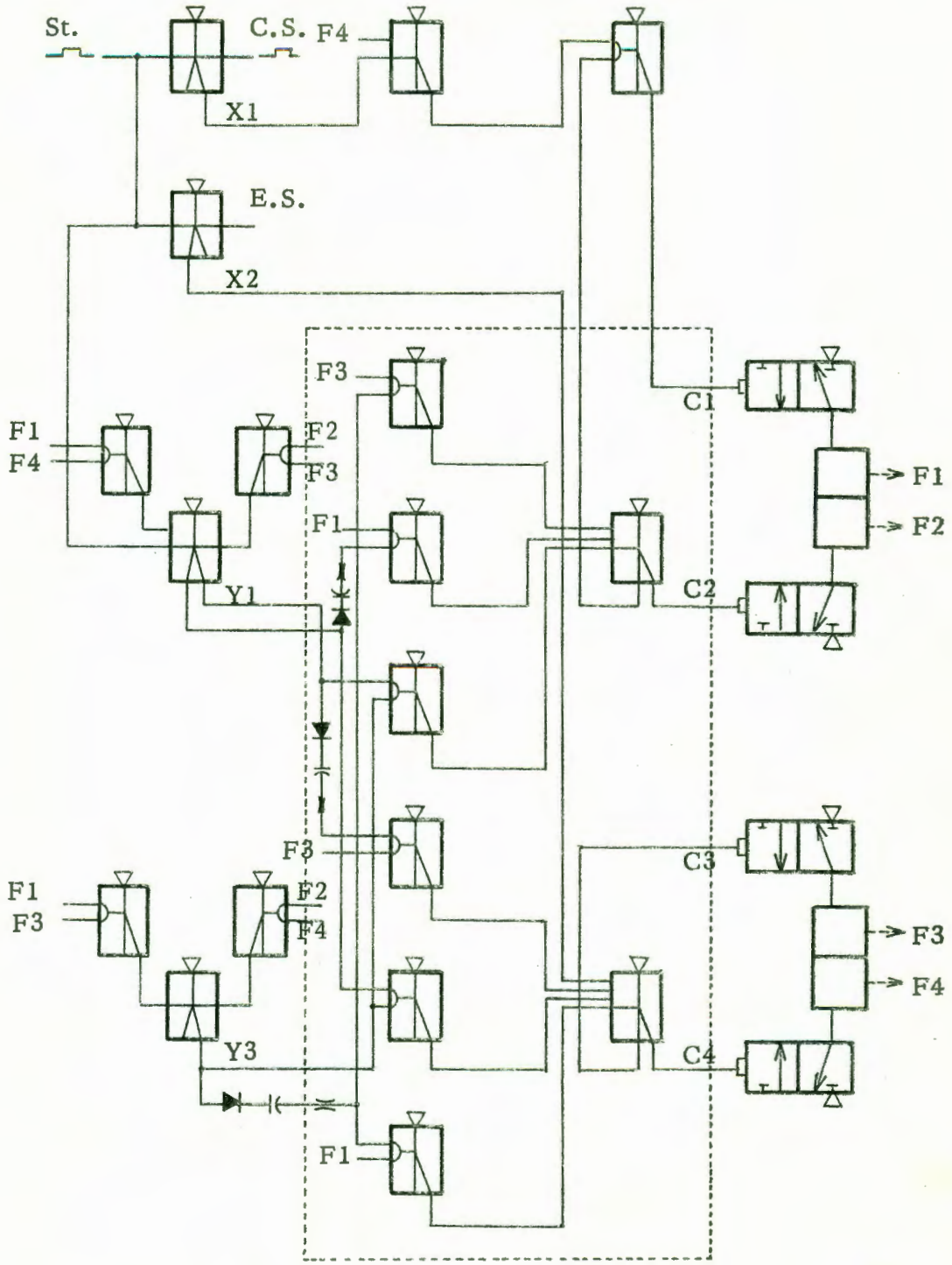


Figure 28. Control circuit diagram of System 2